Frugal GPT-2: Efficient Adaptation via LoRA, Quantization, and Synthetic Data

Repo: `https://github.com/ehersch/cs224n_gpt`

Ryan D'Cunha[†] **Ethan Hersch**[†] **Abhinav Chinta**[†]
Department of Computer Science
Stanford University
{rdcunha, ehersch, achinta}@stanford.edu
[†]Equal contribution

Abstract

While the paradigm of large-scale compute and data has driven unprecedented advancements in natural language processing, full-parameter fine-tuning and data remains costly. This requires highly optimized adaptation strategies for resource-constrained settings. In this work, we investigate the empirical trade-offs between downstream performance and resource utilization during the fine-tuning of GPT-2. We first establish full-parameter baselines across sentiment classification, cloze-style paraphrase detection, and sonnet generation. Building upon sonnet generation, we evaluate three methods of training efficiency: (1) parameter efficiency, mapping performance using Low-Rank Adaptation (LoRA), (2) memory efficiency, analyzing the memory footprint and performance of precision scaling and Quantization-Aware Fine-Tuning (QAFT), and (3) data efficiency, formulating a cost-versus-performance evaluation and robustness for synthetic data augmentation utilizing the Gemini 2.5 family (Flash Lite, Flash, and Pro). We find success in LoRA fine-tuning, evidence of overfitting with quantized-aware fine-tuning, and capacity limits of GPT-2 when performing synthetic data distillation.

1 Introduction

Generative Pretrained Transformer (GPT) models leverage massive corpora of text to pre-train a decoder-only architecture, establishing a foundational "pre-train, fine-tune" paradigm. While GPT-1 [1] proved the viability of this approach for diverse natural language processing (NLP) tasks, GPT-2 demonstrated that scaling parameters (up to 1.5 billion) and training tokens yields dramatic improvements in modeling complex linguistic patterns [2] [3]. However, adapting these architectures to specific, real-world applications exposes critical bottlenecks. The standard practice of full-parameter fine-tuning in high precision imposes prohibitive computational and memory burdens, restricting accessibility for resource-constrained environments [4]. Addressing these inefficiencies is critical to democratizing NLP. We must be able to fine-tune these architectures for downstream tasks while understanding the cost and efficiency of parameters, model precision, and data [5].

The GPT-2 architecture relies on a complex stack of masked multi-head self-attention mechanisms and multi-layer perceptrons (MLPs) [6]. Reducing resource consumption without degrading model capability is notoriously difficult, as naive efficiency approaches result in structural failure [7]. Large-scale language models (LMs) need sufficient data and compute, but overfitting on small, specialized tasks and datasets leads to catastrophic forgetting [8]. On the other hand, aggressive quantization of model weights or lack of data in fine-tuning leads to poor performance on the defined task.

Here, we implement and fine-tune GPT-2 models for downstream tasks of sentiment classification, paraphrase detection, and sonnet generation. We further investigate open-ended sonnet generation to analyze performance impacts across efficiency techniques. We focus on three research questions:

- **Parameter Efficiency**: Can we match performance of full fine-tuning using LoRA while investigating the effects of rank, α, and modules fine-tuned? We find fine-tuning on attention and MLP layers matches full fine-tuning performance with extreme parameter efficiency.
- **Quantization Efficiency**: How does memory footprint through model quantization affect performance with respect quantization during fine-tuning or inference? We observe similar performance with inference quantization and quantized-aware fine-tuning (QAFT) while reducing memory footprint, but we find evidence of overfitting from testing zero-shot performance on other tasks.
- **Data Efficiency**: How does adding synthetic data change model performance across the Gemini-2.5 family? We find improved performance with synthetic data up to a point where GPT-2 capacity limitations prevent improvement with complex models like Gemini 2.5 Pro.

2 Related Works

2.1 Parameter-Efficient Fine-Tuning (PEFT)

Updating all language model weights during fine-tuning imposes severe memory and computational overhead. For a simple fine-tuning approach, adapter modules reduce the number of trainable parameters but introduce inference latency [7]. To address this, Low-Rank Adaptation (LoRA) [9] injects trainable rank decomposition matrices into the existing architecture during post-training, eliminating inference latency. While LoRA is widely adopted, the efficacy across different architectural components, optimal rank, and optimal scaling factor (α) are task-dependent.

2.2 Model Quantization

Open source models can be run on local machines but consume lots of RAM. For instance, downloading Qwen3-14B with FP-16 quantization occupies 30 GB of RAM [10]. Quantization is a powerful tactic to reduce the memory overhead and speed up inference; this is achieved by reducing model parameters to lower precision formats to optimize speed and memory [11]. Post training quantization (PTQ) converts pre-trained weights to lower precision [12], but it can reduce accuracy on inference [13]. However, this may cause a model to underperform as pretrained weights lose precision. Thus, QAFT quantizes weights and performs fine-tuning so that adapted weights can learn respective activations and improve inference [14].

2.3 Synthetic Data Augmentation and Scaling Laws

The performance of LMs scales predictably with model size, compute budget, and dataset volume [3]. However, acquiring the massive, high-quality datasets is often prohibitively expensive. The field has adopted synthetic data generated by frontier LLMs to train smaller, more efficient student models [15] [16]. Despite these advantages, there is a risk of the student overfitting to the teacher [17]. Prior studies validate the utility of synthetic data; we evaluate different-sized synthetic datasets from the Gemini 2.5 family and observe the capacity limitations for GPT-2 to distill from larger models.

3 Approach

3.1 Baseline GPT-2 and Experiments

Our primary baseline for paraphrase detection and sonnet generation leverages the standard GPT-2 Small model with 124M parameters and 12 transformers layers [2], initialized with official pre-trained weights from Hugging Face [18]. We utilize this as its lower FLOP requirements and reduced compute enable rapid iteration for our ablations. For our final evaluation on paraphrase detection, we scale to GPT-2 Large to maximize performance. All extension explorations are limited to open-ended sonnet generation. We impose this scope constraint because highly structured, multi-token generation is a significantly more rigorous stress test for model degradation than a classification task.

3.2 Low-Rank Adaptation (LoRA)

To investigate parameter-efficient fine-tuning (PEFT), we implement LoRA. LoRA posits that the weight updates required for task adaptation reside in a low-dimensional subspace, allowing us to

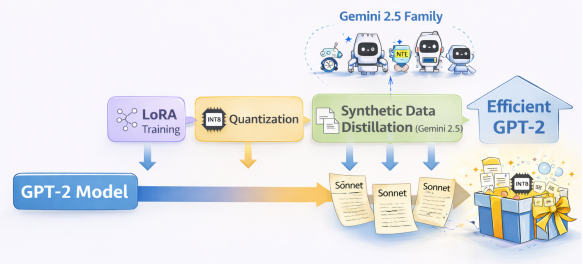

Figure 1: Efficiency explorations for improving a small GPT-2 model. LoRA, quantization, and synthetic data augmentation are investigated to reduce memory and cost while improving performance.

restrict the update space while preserving performance [9]. For a pre-trained, frozen weight matrix $W_0 \in \mathbb{R}^{d\times k}$, LoRA constrains the learned update ΔW via a low-rank factorization:

$$W = W_0 + BA$$

where $B \in \mathbb{R}^{d\times r}$, $A \in \mathbb{R}^{r\times k}$, and rank $r \ll \min(d, k)$. During adaptation, only A and B are trained, while W_0 is frozen. This formulation drastically reduces the trainable parameter count. As rank $r \to \min(d, k)$, LoRA increasingly approaches full fine-tuning in both capacity and computational cost. In practice, LoRA is applied with a scaling factor α such that

$$h = Wx = W_0x + \frac{\alpha}{r}BAx,$$

where $\alpha \in \mathbb{R}$ controls the magnitude of the low-rank update and is often $2r$ [19]. This scaling stabilizes training by ensuring the update remains comparable in scale to frozen pretrained weights [9]. We vary rank, α, and modules trained with LoRA to study its effect on adaptation performance.

3.3 Model Quantization

We start with a baseline full fine-tuned GPT-2 model and quantize it with FP16, BF16, INT8, INT4. We measure memory footprint, tokens per second, and performance. To investigate the impact of quantization on training, we explore the effect of QAFT on sonnet generation performance with INT8 and INT4. We also measure how much QAFT may overfit or lose zero-shot capabilities by evaluating performance on paraphrase detection after being trained for sonnet generation.

3.4 Synthetic Data Augmentation

We then experiment with distillation from frontier models through synthetic data. Due to the limited size of the dataset for sonnet generation (143 training examples) discussed below, we prompt models in the Gemini 2.5 family [20] (Flash Lite, Flash, and Pro) using the Google Vertex API to each generate 1000 Shakespearean sonnets to add significant in-distribution data. Our prompt is shown in Listing 1. We evaluate every generated sonnet to check for rhyming scheme and line count using the `nltk` and `pronouncing` libraries [Table 4]. We find the Gemini models reliably produce the correct 14-line structure, although some generated sonnets deviate from the exact Shakespearean rhyme scheme. Some non-perfect rhymes are generated; also `pronouncing` does not cover all rhymes in the dictionary. The Gemini 2.5 family of models ranges from Flash Lite (0.77B parameters), which is optimized for speed, to Gemini 2.5 Flash and Pro, which are MoE models with enhanced reasoning [21]. Thus, we analyze cost and performance of GPT-2 as we progressively use higher-quality data for distillation. For each model's synthetic data, we fine tune on subsets ranging in size 100, 500, 1000 to measure how much synthetic data is required for performance improvements.

4 Experiments

4.1 Data

We utilize four datasets for training and evaluation with dataset statistics provided in the appendix: the Stanford Sentiment Treebank (SST), comprising 11,855 movie reviews with 5 sentiment labels

(ranging from 0 for Negative to 4 for Positive) 7; the CFIMDB dataset, containing 2,434 movie reviews for binary sentiment classification 6; the Quora dataset, consisting of 404,298 labeled question pairs for paraphrase detection 8; and a corpus of 154 Shakespearean sonnets for open-ended generation. We further add 1000 generated sonnets from the Gemini-2.5 family as discussed 9. For sonnet generation, we are limited by the fixed dev set of test set of 12 sonnets. We preface our results by acknowledging the limitations of our dataset and how an individual incorrectly-generated sonnet can greatly impact final performance.

4.2 Evaluation Method

For evaluation, we focus on task-specific metrics and baseline comparisons to assess model performance. For sentiment classification, we use accuracy as the primary metric, measuring the percentage of correctly classified sentences across sentiment classes in SST and CFIMDB datasets. For paraphrase detection, we use accuracy to measure the model's ability to identify sentence pairs as paraphrases or non-paraphrases correctly in Quora dataset. For the sonnet corpus, we use the chrF metric [22], which is a character level n-gram comparison metric similar to BLEU.

4.3 Experimental Details

For our experiments on the three default project tasks, we utilized Modal and Google Cloud Platform with NVIDIA T4 and L4 GPUs. Our training configurations remained consistent across all experiments using the smallest GPT-2 model, with learning rates ranging from 1e-3 to 1e-5 and batch sizes of 4, 8, 16, 32 based on GPU memory constraints. We train for up to 20 epochs using dev set loss as validation loss to prevent overfitting. For open-ended sonnet generation, we implemented a new generate function. The model first applies temperature scaling ($T = 0.8$) to the output logits. To balance structural coherence with generation diversity, we use a hybrid approach that sequentially applies top-k truncation ($k = 40$) and top-p nucleus sampling ($p = 0.9$).

4.4 Results

Table 1: Baseline performance across all tasks for validation (dev) and test sets. Sentiment classification and paraphrase detection use accuracy, and open-ended sonnet generation uses chrF score.

Task (Dataset)	Fine-Tuning Method	Metric	Dev Score	Test Score
Sentiment (SST)	Last Linear Layer	Accuracy	0.487	0.476
	Full Fine-Tuned	Accuracy	**0.513**	**0.546**
Sentiment (CFIMDB)	Last Linear Layer	Accuracy	0.865	—
	Full Fine-Tuned	Accuracy	**0.971**	—
Paraphrase (Quora)	Full Fine-Tuned	Accuracy	**0.911**	**0.891**
Sonnet Generation	Full Fine-Tuned	chrF	41.974	41.078
Sonnet Generation	Best LoRA	chrF	42.158	—
Sonnet Generation	Best Model Quantization	chrF	42.118	—
Sonnet Generation	Best Data Augmentation	chrF	**46.605**	**52.838**

Currently, our submission (LeBron) is first on the leaderboard for Sonnet Test with a score of 52.838 and 26th for Paraphrase Test with a score of 0.891.

4.4.1 Baseline Results

For all baseline experiments (sentiment classification, paraphrase detection, sonnet generation), we fully fine-tune GPT-2. We perform hyperparameter grid search on learning rate and batch size to obtain the highest performing models. For sentiment classification, we also provide results fine-tuning only the last layer of the network. All results are shown in Table 1. Downstream extension experiment best results are also shown, with test set performance only reported when present and limited due to 3 maximum submissions for a test score. For sentiment classification, full fine-tuned models had higher performance than their last linear layer fine-tuned counterparts which is expected as more task specific characteristics can be learned by the network as more parameters are allowed to be updated.

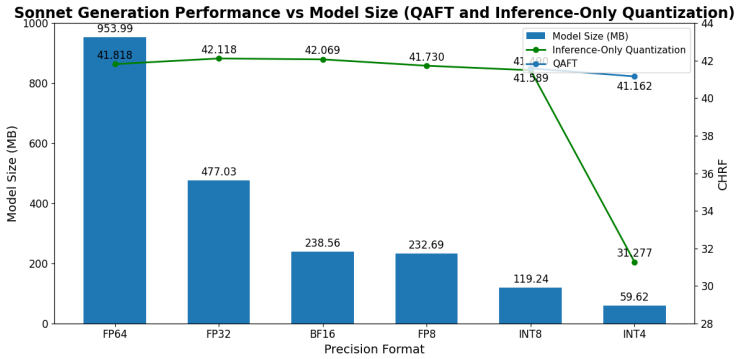

Figure 2: Sonnet generation performance relative to model size using quantization on GPT-2. Inference-only quantization applies post-training weight-only quantization to a fully fine-tuned sonnet model. QAFT applies weight quantization during SFT.

4.4.2 LoRA

We explore LoRA for PEFT to determine if lower ranks can match performance of the fully fine-tuned baseline. By sweeping over learning rates from 1e-5 to 5e-2 with LoRA ranks 1 to 256, we find rank 256 with a learning rate of 1e-2 is best performing with a dev chrF as 42.058. A batch size of 16 is used for these experiments due to memory constraints with the large scale of ablations run. All rank results are shown in Table 3 with all dev chrF scores being within 1 of the fully fine-tuned baseline. The table includes the number of trainable parameters with each rank to observe the computational efficiency achieved. For this task, even extreme low ranks can match performance of full fine-tuning. We hypothesize this is because GPT-2 has already learned the vast majority of English syntax, grammar, and vocabulary during pre-training. To adapt it to write sonnets, the model does not need to relearn English, but rather, a simple directional nudge towards sonnet structure.

We fixed α at 16, but with rank 256 and learning rate 1e-2, we then ablate α (scaling factor) with 8, 16, 32, 64, 128, 256, 512 (up to $\alpha = 2r$) to successfully confirm the used $\alpha = 16$ was best [Table 5]. We see that LoRA requires a far higher learning rate than full fine-tuning, which is expected as the limited parameters being updated in LoRA can safely use a higher learning rate to ensure the updates are large enough to shift model predictions. Without a larger test set and bootstrapping we can't claim performance improvement, but we find that with an optimal learning rate and rank, LoRA can at least match a full fine-tuned baseline using fewer fine-tuning parameters for sonnet generation.

With the best rank of 256, learning rate of 1e-2, $\alpha = 16$, we further investigate the impact of where LoRA is applied. We compare LoRA on all modules (attention, attention output, and MLP), versus attention and attention output (no MLP). Training all modules results in a higher dev chrF score of 42.158 compared to the no MLP training chrF score of 41.483 [Table 6] while maintaing a lower train and dev loss. We hypothesize that this is expected as the MLP contains roughly two-thirds of the dense parameters. By applying LoRA only to attention, there are simply not enough parameters being updated to capture the learned sonnet structure.

4.4.3 Model Quantization

In our exploration of quantization, we measure the efficiency we can gain in memory footprint and inference speed on sonnet generation. We measure the amount of space the model occupies in memory (MB) and compare this against the dev set chrF score achieved by these quantized models. As shown in Figure 2, switching to BF16 or FP8 halves the memory footprint of GPT-2 while offering minimal downside on inference. INT8 reduces the model to only 119 MB and INT4 reduces size to 60 MB, but performance drops by over 20% for this quantization. Since the performance drop is between INT8 and INT4, we test QAFT for these precisions, and performance is retained. chrF score only drops from 42.118 to 41.162 from our original FP32 GPT-2 model. Table 2 shows that lower precision formats, on top of providing considerable memory optimizations, also considerably optimize the number of tokens output per second during inference.

Table 2: Inference speed and model size comparison across different numerical precision formats for GPT-2 Small. Inference is performed on one L4 GPU with a batch size of 8.

Variant	Memory Footprint (MB)	Tokens/s
FP32	477.02	78.108
FP16	238.56	85.899
BF16	238.56	82.692
INT8	119.24	100.486
INT4	59.62	103.734

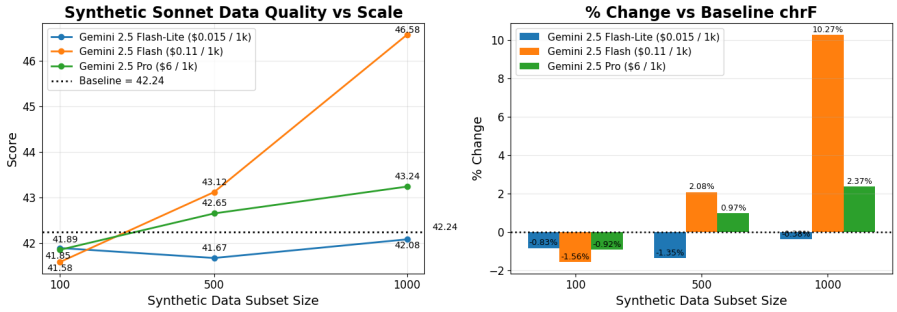

Figure 3: Comparison of the amount of synthetic data to dev CHRF. All subsets run on 20 epochs with a batch size of 32 and a learning rate of $1e-3$. Performing SFT on GPT-2 for sonnet generation.

4.4.4 Synthetic Data Augmentation

We also explore performance improvement using synthetic data from a more complex model. The goal of this experiment is to not only see how synthetic data improves the performance of GPT-2, but also how using higher quality synthetic data across the Gemini 2.5 family affects performance.

In Figure 3 we see when using Gemini 2.5 Flash Lite, adding more data hurts performance since this model produces lower quality data. Flash Lite is optimized for speed and cost, not advanced reasoning or verbosity [23]. Flash is a model more optimized for complex tasks involving more advanced language and understanding (and $100\times$ more expensive). When adding synthetic sonnet data from Flash, accuracy increases from our Baseline SFT performance of 42.243 to 46.60. Finally, we evaluate on Gemini 2.5 Flash Pro data. For SFT on subsets of 100 and 500 examples, its chrF performance is similar to that of 2.5 Flash. However, when fine tuning on all 1000 examples, it only reaches an accuracy of 43.236. It also costs $\sim$ \$6 to generate this data. We would assume that 2.5 Pro would produce the highest-quality in-distribution data and make our model overperform, but we found that its performance plateaus. This could be because Gemini 2.5 Pro is too large a teacher for such a small GPT-2 model, so distillation does not work. We further explore in the analysis.

5 Analysis

5.1 Baseline Experiment Analysis

As observed in the sentiment classification evaluations (SST and CFIMDB), full fine-tuning consistently outperforms linear probing (last layer fine-tuning). This performance gap indicates that the frozen pre-trained representations of GPT-2, while robust, are not perfectly linearly separable for these specific target domains. Furthermore, we find that hyperparameter sweeps are crucial to improving performance across tasks, and each task has different optimal hyperparameters. On small datasets like sonnet generation, we also found it important to include early stopping to prevent overfitting to the training data within early epochs.

5.2 LoRA

While LoRA results were found for the best learning rate at each rank, we further wanted to explore how the loss of LoRA compares to full fine-tuning sweeping over learning ranks for train and dev. We visualize this in Figure 4 with the justified $\alpha = 16$ with LoRA applied to all layers. Since early stopping was part of the training loop for overfitting, we show the train and dev loss at the last epoch. We see that the fully fine-tuned model reaches its lowest loss at a learning rate an order of magnitude smaller than the LoRA results. We hypothesize that LoRA optimizes a significantly lower-

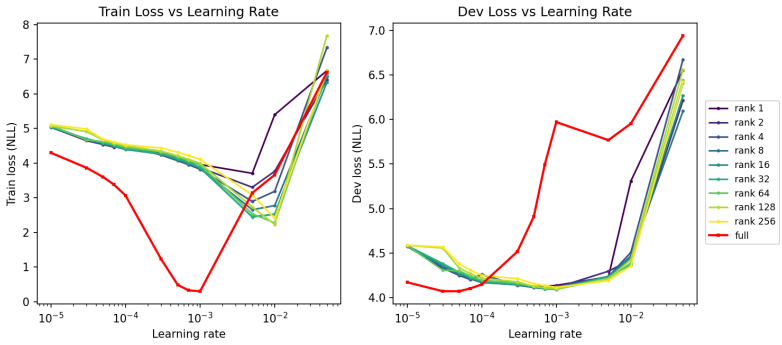

Figure 4: Sonnet generation train and dev loss at the final epoch to account for overfitting across LoRA ranks and learning rates. A full fine-tuned baseline is included as a comparison.

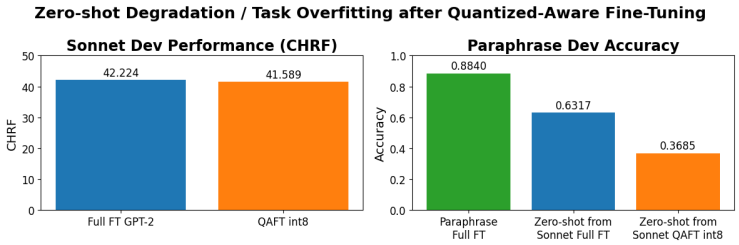

Figure 5: Overfitting of QAFT. GPT-2 Small is quantized to the INT8 format and then full fine tuned on sonnet generation. Its zero-shot performance for the paraphrase detection task is compared. All fine tuning run on 20 epochs with a batch size of 32 and a learning rate of $1e-3$.

dimensional parameter space compared to full fine-tuning, which acts as an implicit regularizer and allows the model to tolerate larger optimization steps without destabilizing the pre-trained weights. We also see that the train loss for fully-fine tuned at 1e-3 learning rate is significantly lower than any LoRA rank train loss, but the dev loss at the same learning rate is very high. This indicates the fully fine-tuned model is overfitting while most LoRA rank models maintain a consistent dev loss over the sweep as it does not possess the ability to fully overfit since not all parameters can be updated. LoRA is more robust to overfitting while being more parameter efficient in the sonnet generation setting.

5.3 Model Quantization

Our quantized models matched relative performance of full precision in Figure 2. The INT4 format is a fraction of the size of FP32, yet retains 98% of chrF score with QAFT. The INT8 model also loses little performance. We hypothesize these QAFT models were overfitting to sonnet generation and losing performance on other tasks. In Figure 5, we perform QAFT on GPT-2 with INT8 precision for sonnet generation. Performance is relatively similar with QAFT compared to full fine-tuning. If we take an un-quantized GPT-2 model with SFT on sonnet generation and evaluate it on paraphrase detection, performance drops $\sim 28\%$. However, if we use the QAFT (INT8) from sonnet completion, zero-shot paraphrase detection performance drops $\sim 58\%$. QAFT has minor affects performance on a fine-tuned task, but it severely affects generalized performance, indicating overfitting.

Quantization reduces the numerical precision of model parameters, introducing approximation errors that can degrade downstream performance. Empirical studies show that low-bit quantization can impact zero-shot performance, as small changes in weights may disrupt the general representations learned during pre-training [14, 24].

5.4 Synthetic Data Augmentation

For synthetic data, our hypothesis was that the model is dramatically overfitting to dev set sonnet generation, leading us to test its zero-shot abilities on paraphrase detection. According to the results in Table 7, even when fine-tuned on 1000 synthetic sonnets, the models do not lose zero-shot performance on dev set paraphrase detection, exhibiting an accuracy of 0.63. As shown before, the model fine-tuned on Gemini 2.5 Flash-Lite data does not perform better than baseline on the sonnet

generation task. However, the model that uses Gemini 2.5 Flash data increases in chrF score by $\sim 10\%$ and loses no zero-shot accuracy. This demonstrates that the model fine-tuned on synthetic data is not overfitting to the sonnet generation task, and it is preserving some general ability to perform other (zero-shot) tasks. While neural networks are highly expressive and capable of memorizing training data, modern empirical results show that overparameterized models can still generalize well despite their capacity to overfit [25]. Our results suggest that synthetic distillation at this scale does not significantly degrade the model's broader language capabilities.

For all three models, performance with 100 synthetic examples is slightly worse than the baseline [Figure 3]. The original training set contains only 143 human-written sonnets, so introducing 100 synthetic samples likely alters the training distribution. This can slightly degrade performance when only a small amount of synthetic data is added.

Even when fine-tuning on a frontier model's data, a smaller student may struggle to fully distill knowledge from a much larger teacher [26, 27]. In our experiments, using all 1000 Gemini 2.5 Pro-generated examples improves chrF performance on the sonnet generation task but does not surpass the performance achieved when distilling from Gemini 2.5 Flash. This non-monotonic behavior suggests that stronger teacher models do not necessarily produce more effective synthetic training data for smaller students. A possible explanation is a capacity mismatch: the complexity of Pro-generated sonnets may be hard for a GPT-2-scale model to imitate. We investigate if this finding exists when using GPT-2 Large. When fine-tuning the student GPT-2 Large model with synthetic Gemini Pro data, we find the distribution captured by the Gemini data still cannot be distilled properly. The chrF score on all 1000 examples is 43.716, indicating no real major improvement GPT-2 Small.

6 Conclusion and Future Work

We evaluate empirical trade-offs across parameter, memory, and data efficiency when adapting GPT-2 under constrained compute and memory budgets. First, we demonstrate that LoRA can match full fine-tuned performance in sonnet generation while reducing trainable parameters and mitigating overfitting. Second, while inference-time quantization degrades performance at lower precisions, QAFT effectively stabilizes performance on sonnet generation. However, it risks task-specific overfitting and reduced zero-shot capabilities for paraphrase detection. Finally, our synthetic data evaluation using the Gemini 2.5 family reveals that while higher-quality data improves distillation, GPT-2's capacity limits prevent full knowledge transfer. Notably, augmenting training with synthetic sonnets preserves zero-shot paraphrase accuracy, suggesting this scale of augmentation does not induce catastrophic overfitting.

Limitations and Future Work

Our study has several limitations that motivate future work. First, our sonnet generation evaluation is constrained by the small size of the human-written training, validation, and test sets. Because the dev and test sets contain only a small number of examples, a single poor generation can substantially affect the final chrF score. Future work should evaluate these efficiency methods on larger open-ended generation datasets and supplement automatic metrics with human evaluation of poetic quality. Additionally, because the held-out test set is not publicly accessible, we cannot directly audit whether synthetic generations contain exact or near-duplicate overlap with test examples. Future work should use explicit deduplication and contamination checks when evaluating synthetic data augmentation.

Second, our experiments focus primarily on GPT-2 Small, with only limited scaling to GPT-2 Large for synthetic data distillation. A natural extension would be to repeat the LoRA, quantization, and synthetic augmentation experiments across a wider model family to better understand how these trade-offs change with model capacity. This would help separate the limitations of the efficiency methods themselves from the limitations caused by the relatively small student model.

Finally, our synthetic data results motivate a more detailed study of teacher-student mismatch. Although Gemini 2.5 Pro produces structurally valid sonnets at a high rate, its data did not yield the strongest downstream GPT-2 performance. Future work could analyze whether this is due to stylistic mismatch, excessive linguistic complexity, or insufficient student capacity. Filtering synthetic examples by rhyme correctness, perplexity, diversity, or teacher confidence may also improve the cost-performance tradeoff of synthetic augmentation.

References

[1] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 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] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[4] Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in nlp. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 3645–3650, 2019.

[5] Roy Schwartz, Jesse Dodge, Noah A Smith, and Oren Etzioni. Green ai. *Communications of the ACM*, 63(12):54–63, 2020.

[6] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[7] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp, 2019.

[8] Hongyu Li, Liang Ding, Meng Fang, and Dacheng Tao. Revisiting catastrophic forgetting in large language model tuning. In *Findings of the association for computational linguistics: EMNLP 2024*, pages 4297–4308, 2024.

[9] 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, 2021.

[10] ApXML. Qwen3-14b model specifications. `https://apxml.com/models/qwen3-14b`, 2025. Accessed: 2026-03-06.

[11] Shiyao Li, Xuefei Ning, Luning Wang, Tengxuan Liu, Xiangsheng Shi, Shengen Yan, Guohao Dai, Huazhong Yang, and Yu Wang. Evaluating quantized large language models, 2024.

[12] Jinjie Zhang, Yixuan Zhou, and Rayan Saab. Post-training quantization for neural networks with provable guarantees. *SIAM journal on mathematics of data science*, 5(2):373–399, 2023.

[13] Yoni Choukroun, Eli Kravchik, Fan Yang, and Pavel Kisilev. Low-bit quantization of neural networks for efficient inference. *arXiv preprint arXiv:1902.06822*, 2019.

[14] Quan Wei, Chung-Yiu Yau, Hoi-To Wai, Yang Katie Zhao, Dongyeop Kang, Youngsuk Park, and Mingyi Hong. Roste: An efficient quantization-aware supervised fine-tuning approach for large language models, 2025.

[15] Anup Shirgaonkar, Nikhil Pandey, Nazmiye Ceren Abay, Tolga Aktas, and Vijay Aski. Knowledge distillation using frontier open-source llms: Generalizability and the role of synthetic data, 2024.

[16] André Bauer, Simon Trapp, Michael Stenger, Robert Leppich, Samuel Kounev, Mark Leznik, Kyle Chard, and Ian Foster. Comprehensive exploration of synthetic data generation: A survey, 2024.

[17] Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. Orca: Progressive learning from complex explanation traces of gpt-4. *arXiv preprint arXiv:2306.02707*, 2023.

[18] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

[19] Hugging Face. Hugging face llm course: Chapter 11, 2024. Accessed: 2026-03-10.

[20] Google DeepMind. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

[21] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities, 2025.

[22] 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.

[23] Google DeepMind. Gemini 2.5 flash-lite model card. `https://deepmind.google/models/model-cards/`, 2025. Updated September 26, 2025. Accessed March 7, 2026.

[24] Ron Banner, Yury Nahshan, Elad Hoffer, and Daniel Soudry. Post-training 4-bit quantization of convolution networks for rapid-deployment, 2019.

[25] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization, 2017.

[26] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015.

[27] Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes, 2023.

```
"""
Generate 1000 Shakespearean sonnets in English, in the style of Shakespeare's
    original sonnets.

Hard constraints for EACH sonnet:
- Exactly 14 non-empty lines.
- No title line.
- Strict end-rhyme scheme: ABAB CDCD EFEF GG (clear end rhymes; avoid slant/near
     rhymes).
- Shakespearean diction and syntax (Early Modern English feel; no modern slang).

Formatting constraints:
- Before each sonnet, output a single line containing exactly: ###{
    starting_index}### for the first sonnet, then increment by 1 for each next
    sonnet.
- After the delimiter, output the 14 lines of the sonnet.
- Output nothing else.

Begin.
"""
```

Listing 1: The prompt template used for generating Shakespearen sonnets from Gemini-2.5 models.

Table 3: LoRA rank and validation chrF score for sonnet generation. For each rank, we report the optimal learning rate that yielded the highest dev chrF score using a constant batch size of 16 and $\alpha = 16$ as well as the trainable parameters in each setting. Learning rate was swept over between 1e-5 and 5e-2. A full fine-tuning baseline is included.

Configuration	**Trainable Parameters**	**Optimal Learning Rate**	**Dev chrF Score**
LoRA Rank 1	165,888	5×10^{-3}	41.679
LoRA Rank 2	331,776	1×10^{-3}	41.763
LoRA Rank 4	663,552	3×10^{-5}	41.369
LoRA Rank 8	1,327,104	1×10^{-3}	41.927
LoRA Rank 16	2,654,208	5×10^{-4}	41.777
LoRA Rank 32	5,308,416	1×10^{-3}	42.008
LoRA Rank 64	10,616,832	5×10^{-4}	41.876
LoRA Rank 128	21,233,664	5×10^{-3}	42.020
LoRA Rank 256	42,467,328	1×10^{-2}	**42.158**
Full Fine-Tuning	$\sim$124M	7×10^{-5}	41.9093

Table 4: Percentage of generated sonnets that match the Shakespearean rhyme scheme (ABAB CDCD EFEF GG) using CMU Pronouncing Dictionary rhyme detection and are 14 lines.

Model	**Perfect Sonnet Rate (%)**
Gemini 2.5 Flash Lite	56%
Gemini 2.5 Flash	72%
Gemini 2.5 Pro	75%

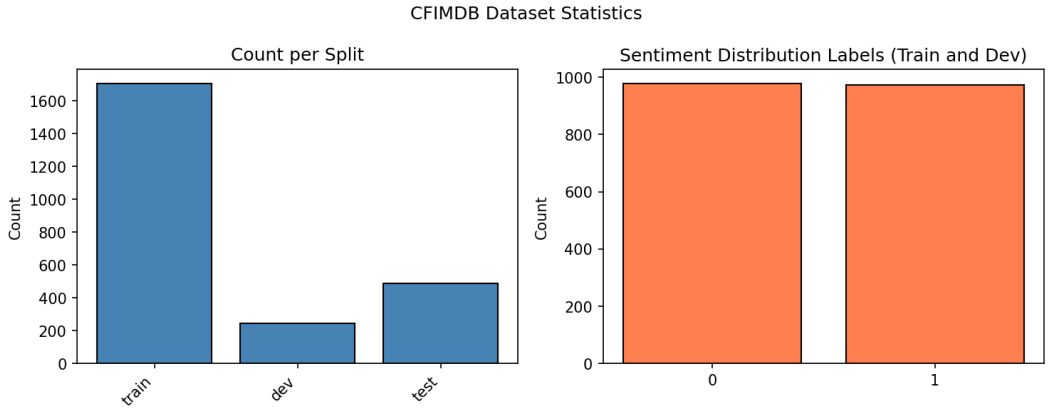

Figure 6: Dataset statistics for CFIMDB used for sentiment classification. A train, dev, and test split was used with labels shown for train and dev label distributions.

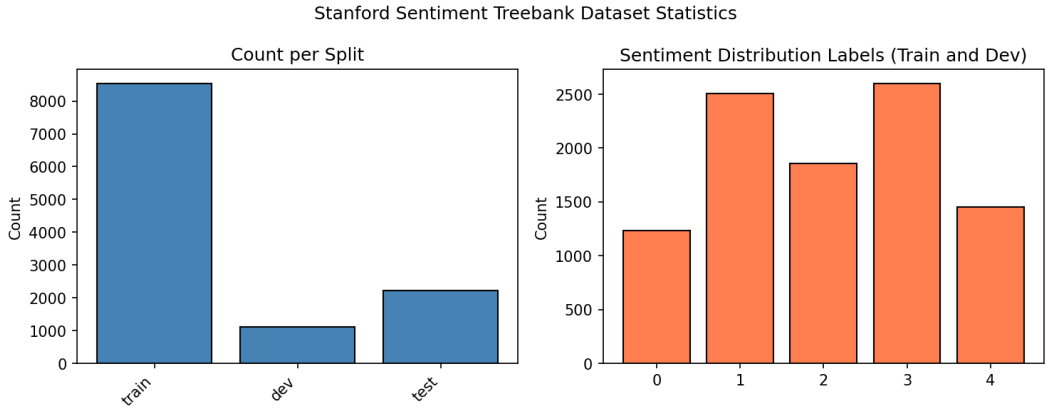

Figure 7: Dataset statistics for SST used for sentiment classification. A train, dev, and test split was used with labels shown for train and dev label distributions.

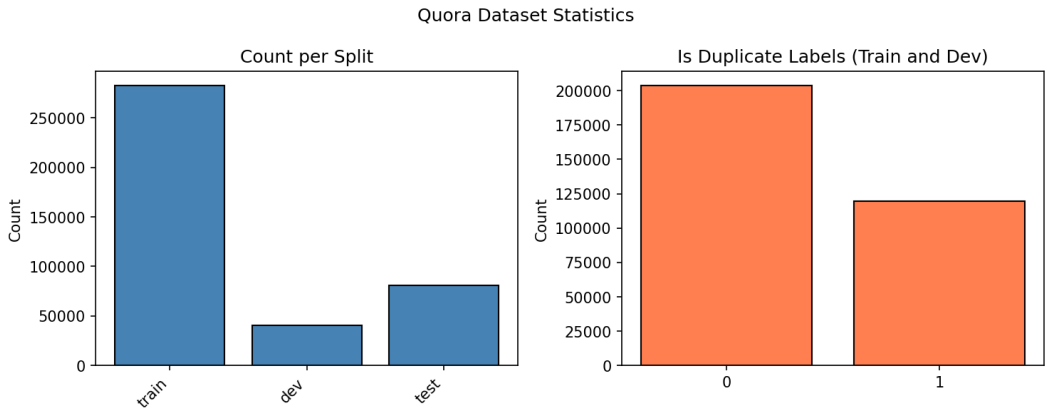

Figure 8: Dataset statistics for Quora used for paraphrase detection. A train, dev, and test split was used with labels shown for train and dev label distributions.

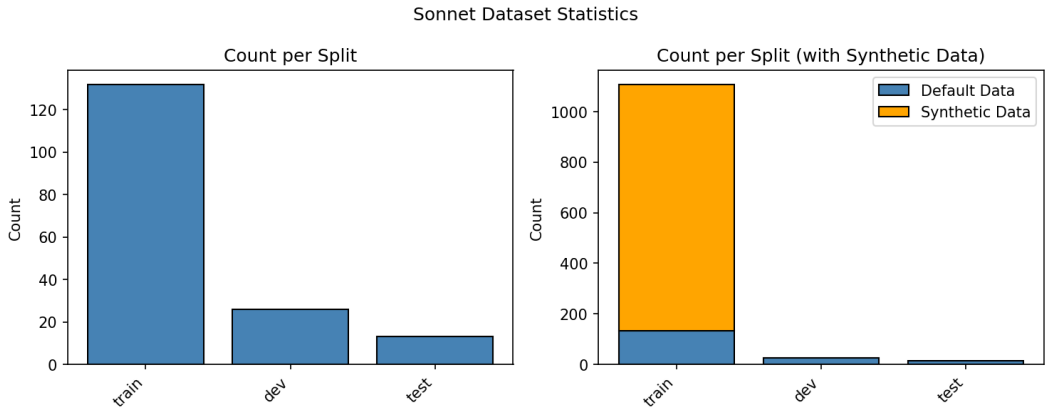

Figure 9: Dataset statistics for sonnets used for paraphrase detection. A train, dev, and test split was used with synthetic data added in for training.

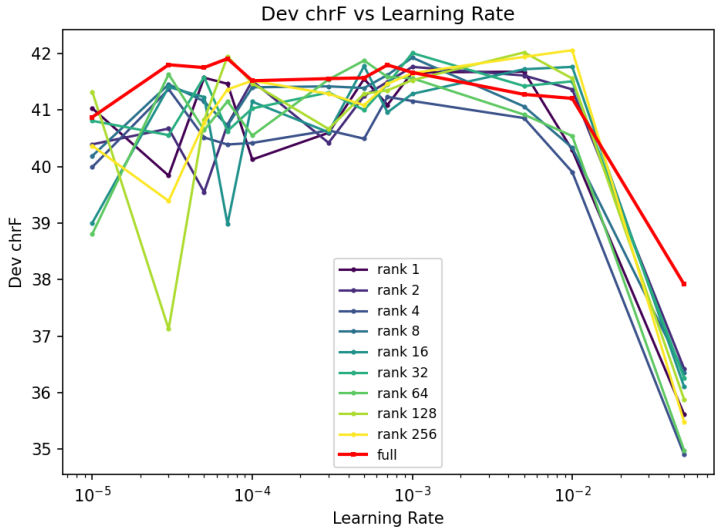

Figure 10: Sonnet generation dev chrF score for LoRA ranks across learning rates. A full fine-tuned baseline is included. Rank 256 has the best dev chrF performance of 42.058 at a learning rate of 1e-2

Table 5: Impact of the LoRA scaling factor (α) on dev chrF performance for sonnet generation. Hyperparameters were based on previous ablations with batch size 16, learning rate 1e-2, rank 256.

α Scaling Factor	**8**	**16**	**32**	**64**	**128**	**256**	**512**
Dev chrF	41.824	**42.158**	41.777	40.041	35.531	36.087	37.057

Table 6: Ablation study of targeted LoRA modules. Restricting LoRA to the attention mechanism and freezing the Multi-Layer Perceptron (MLP) results in a measurable drop in validation performance. All layers represents the attention, attention output, and MLP layers.

Modules Trained	**Train Loss**	**Dev Loss**	**Dev chrF Score**
All (Attention + MLP)	2.404	4.170	42.158
No MLP (Attention Only)	3.457	4.355	41.483

Table 7: Zero shot capabilities of SFT sonnet generation with a baseline model fine-tuned on provided sonnet data and models fine-tuned with additional synthetic data evaluated on paraphrase detection. All models exhibit similar paraphrase accuracy despite 2.5 Flash SFT's superior sonnet generation.

Model	**Sonnet Generation Dev chrF**	**Paraphrase Dev Zero Shot Accuracy**
Sonnet Generation SFT (Baseline)	42.243	0.6317
2.5 Flash SFT (1k)	46.600	0.6317
2.5 Flash-Lite SFT (1k)	42.092	0.6315