

Low-Rank Adaptation and Preference Optimization for Accessible Multi-Task GPT-2 Fine-Tuning

Stanford CS224N {Default}

Mahathi Mangipudi
Department of Computer Science
Stanford University
mahathim@stanford.edu

Taylor Hamilton-Hankins
Department of Computer Science
Stanford University
thamhank@stanford.edu

Tyler Ho
Department of Computer Science
Stanford University
tylerho@stanford.edu

Abstract

Fine-tuning large language models (LLMs) enables strong task-specific performance but often incurs substantial computational cost, particularly for models with hundreds of millions to billions of parameters. GPT-2 architectures, for example, range from 117 million to 1.5 billion parameters, making full fine-tuning resource-intensive. We explore how parameter-efficient fine-tuning (PEFT) methods can reduce computational overhead while preserving downstream performance. We apply Quantized Low-Rank Adaptation (QLoRA) paraphrase detection and compare its performance with full fine-tuning under the accuracy metric. We assess QLoRA's performance relative to the number of trainable parameters. Additionally, we implement Direct Preference Optimization (DPO) to further enhance sonnet generation performance under the CHRF metric.

1 Key Information to include

- Mentor: Advit Deepak
- External Collaborators (if you have any): None
- Sharing project: No
- Late Days (if you are using any):

Team member	Late days remaining
<Mahathi Mangipudi>	0
<Taylor Hamilton-Hankins>	0
<Tyler Ho>	3
Total pooled	3
Team size	3
Late days applied for the report	1

2 Introduction

2.1 QLoRA

LLMs have gained recognition for their versatility and adaptability across a wide range of tasks. However, achieving this level of performance typically requires models with billions of parameters,

making full fine-tuning computationally expensive. Training requires substantial GPU memory, compute time, and energy, and in addition to the resources themselves being limited, access to them is constrained by computational infrastructure. This creates a practical bottleneck because progress and experimentation can become restricted by the cost and availability of these resources. Consequently, a key challenge in the modern age of machine learning is maintaining or improving model performance while reducing the computational cost required for training and adaptation.

Beyond immediate cost constraints, this problem is interesting because it addresses the boundary between theoretical capability and practical deployment. In theory, LLMs can perform a vast amount of tasks across a variety of domains and perform well. However, in practice, computational costs limit the extent to which these possibilities can be explored leaving many ideas and applications unrealized. Therefore, mollifying the computational burden of the training and experimentation process has implications that extend beyond cost alone. It would expand the range of researchers who can work on these problems and enable more rapid and pervasive exploration of new ideas and domain-specific applications.

As we would expect, the difficulty of this problem is proportional to the weight of its impact. Though many advancements have been made in this research domain, some fundamental attributes of the models and the training process make this a difficult problem to address. Model performance is owed in large part to the sheer number of parameters and their expressive capability. Reducing the number of trainable parameters can easily atrophy the capability of the model if not done well. We also must store model weights, gradients, optimization coefficients, and activations when training transformer-based architectures making the process quite demanding even for smaller adaptations. Any approaches to this problem must balance efficiency and stability, and evaluation of these approaches is an expensive process in and of itself as full fine-tuning and training are often used as benchmarks.

Despite a large subset of research still being dedicated towards overcoming the performance - computational cost tradeoff, there have been many successful advancements in this domain, including parameter-efficient fine-tuning (PEFT) techniques, quantization, model compression, and more, that are used individually and in combination. These methods manipulate precision, parameter storage, and computational complexity in order to reduce computational cost. In this project, we focus on implementing a combination of quantization and parameter-efficient fine-tuning, more specifically QLoRA to streamline training and maintain / improve performance of GPT-2 on the downstream task paraphrase detection.

2.2 DPO

Also motivated by model performance, another issue we face is controllability and model alignment with human preferences. Ideally, LLM performance on a specified task should meet or exceed human level performance according to human preference based evaluations. However, the standard training objective for these models is predicting the most likely next token. This does not directly optimize for alignment with human preferences on task specific expectations. As a result, even highly capable models can produce results that diverge from desired behavior. Because training is largely unsupervised, and the models contain billions of parameters, it is difficult to exert precise control over how they respond in different contexts or reliably guide their behavior towards preferred outcomes. Consequently, stronger performance on conventional evaluation metrics does not necessarily guarantee that model outputs will align with human judgments of quality or usefulness. Therefore, another key problem in modern LLM development is effectively aligning model behavior with human preferences while maintaining the underlying capabilities learned during training.

Ensuring that LLMs align with human judgment is an important problem because model capability alone does not guarantee practicality or usefulness. Outputs are only valuable if they correspond with human evaluations of quality, and even with advanced reasoning ability, model outputs can vary widely in quality, relevance, and appropriateness when applied to real world tasks. It is here where the problem becomes interesting as LLMs are designed to generate responses based on learned patterns that may not be apparent to the human eye. Bridging the gap between controllability and reliability is essential for translating the underlying capabilities of LLMs to systems intended to assist users, provide information, or generate content.

This problem of alignment is challenging both because of the data used to perform alignment and because of constraints inherent to these models. Human preferences vary widely due to their subjective

and context-dependent nature, making it difficult to define a single objective that consistently captures preferred behavior. Furthermore, human preferences can be costly to collect at scale. On the model side, generation is probabilistic, so alignment methods must shape the distribution of possible responses rather than enforce a single deterministic behavior. This poses a challenge to consistency across model outputs given different prompts and sampling conditions.

RLHF typically collects human judgments to train a reward model [1, 2]. However, as human annotations are expensive, and evaluators may inconsistently apply scoring standards, this process can be costly and difficult to scale. RLHF does so relatively effectively by training a separate reward model based on human preferences and then using that model as a reward function to optimize policy through reinforcement learning. Though effective, RLHF is not very efficient due to the additional components. DPO directly updates the model’s policy in a preliminary supervised fine-tuning stage, streamlining the optimization process. We focus on DPO in this project to improve performance of GPT-2 on the downstream task sonnet generation.

3 Related Work

The transformer architecture and its self-attention mechanism enables modern large language models (LLMs) development [3]. Such models demonstrate strong few-shot and even zero-shot generalization capabilities, as illustrated by models such as OpenAI’s GPT-2 [4]. However, performance often increases with supervised fine-tuning (SFT). Fully fine-tuning all parameters can be computationally expensive and often infeasible. Consequently, parameter-efficient fine-tuning (PEFT) techniques seek to minimize computational requirements while maintaining performance [5]. Low-Rank Adaptation (LoRA) is a PEFT method that introduces trainable low-rank matrices into selected layers of a pretrained model [6, 7, 8]. Instead of FFT, LoRA constrains updates to these low-rank components, thereby substantially reducing the number of trainable parameters while maintaining strong task performance.

LLMs’ increasing scale introduces substantial memory requirements. To address this challenge, prior work has explored weight quantization, reducing higher-precision representations such as 32-bit or 16-bit floating-point weights to lower-precision formats including 8-bit or 4-bit representations, which can decrease model storage cost by several-fold [9, 10, 11, 12]. Building on these advances, Quantized Low-Rank Adaptation (QLoRA) combines low-bit quantization of pretrained model weights with LoRA updates [13]. This approach reduces base model memory footprint and restricts training to low-rank adaptations within selected layers, thereby significantly lowering SFT’s computational and memory costs.

Users often want LLM-generated text to satisfy specific criteria, be it factuality, conciseness, or creativity. To better align with human goals, reinforcement learning from human feedback (RLHF) is widely used for post-training alignment [1, 2]. RLHF typically collects human judgments to train a reward model. However, as human annotations are expensive, and evaluators may inconsistently apply scoring standards, this process can be costly and difficult to scale. Direct Preference Optimization (DPO) addresses these limitations by learning directly from pairwise preference comparisons [14]. It aligns models to human preferences by optimizing for the preferred responses and minimizing rejected responses.

4 Approach

We fine-tuned models from the GPT-2 family of transformer architecture on two downstream tasks: paraphrase detection and sonnet generation [4]. Let d denote the model’s hidden dimension size.

4.1 Baseline

For the paraphrase detection task, we fine-tuned our pretrained GPT-2 model on the Quora Questions dataset by framing paraphrase detection as a binary classification problem. Given two input questions, our model predicts whether they express the same semantic meaning. Specifically, instead of modifying the transformer architecture, we leverage GPT-2 language modeling by using a prompt-based approach to detect paraphrasing. We format each example into a yes or no question using a

natural language prompt: Is "<s1>" a paraphrase of "<s2>"? Answer "yes" or "no": where s_1 and s_2 denote the two input questions respectively.

The prompt is passed through the GPT-2 transformer, which produces contextualized hidden representations for each token in the sequence. We extract the hidden representation corresponding to the final token, which should be either a "yes" or a "no," and project it to vocabulary logits using the pretrained modeling head for GPT-2.

In our baseline, we performed full fine-tuning of the GPT-2 small model using the AdamW optimizer with default hyperparameters such as a batch size of 8 and a learning rate of $1e-5$. This baseline achieved 89.9% accuracy on the development set and 87.4% accuracy on the test set.

To perform sonnet generation, we formulate an autoregressive language task using GPT-2 in its traditional sense. The model predicts the next token ($P(x_{t+1}|x_{\leq t})$) based on the tokens seen prior ($x_{\leq t}$), and uses this process repeatedly to generate text token by token. We initialize GPT-2 with pretrained weights and fine-tune it on a corpus of Shakespeare’s sonnets using the standard next-token prediction objective. During generation, we condition the model on the first three lines of selected sonnets and allow the model to complete the remaining lines autoregressively.

We pass the final hidden states of the transformer through a position-wise projection MLP, consisting of LayerNorm, a linear expansion from d to $4d$, GELU activation, dropout, and a linear projection back to d . The resulting projections are then mapped to vocabulary logits via the language modeling head.

$$\tilde{H} = W_2 \text{Dropout}(\text{GELU}(W_1 \text{LayerNorm}(H))), \quad \text{where } W_1 \in \mathbb{R}^{4d \times d}, W_2 \in \mathbb{R}^{d \times 4d} \quad (1)$$

The expansion-compression structure increases the expressive capacity of the output by enabling nonlinear transformations in a higher-dimensional feature space, while maintaining compatibility with the vocabulary projection layer. The design follows the feed-forward structure commonly used in transformer blocks, which has proven effective for modeling complex representations. We use full fine-tuning to meet baseline performance (CHRF = 41.525).

4.2 QLoRA Implementation

Models were quantized using the bitsandbytes library by replacing PyTorch linear layers with their 4-bit or 8-bit quantized equivalents. LoRA was implemented using the HuggingFace peft library, creating low-rank adapters on the attention mechanism projection matrices W_k and W_v following [6]. Under QLoRA, the model weights are quantized and frozen, and task-specific updates are learned through LoRA adapters.

4.3 Direct Preference Optimization Implementation

To help nudge the sonnets generated by our model to the gold standard of the original Shakespearean, we constructed several sets of preference pairs for use with Direct Preference Optimization. DPO requires pairs of a preferred output and a less desirable outputs for the same prompt. Our chosen and rejected sonnets were generated in a three step pipeline where we first prompted moonshot-v1-8k model via the Moonshot AI Kimi API, filtered generated sonnets by whether the starter lines were copied exactly and if the sonnet had exactly 14 lines, before using manual review and human annotations to fix those issues and ensure the pairs best encode our preferences.

Ultimately, we constructed a few variants of the DPO preference pairs based on two primary characteristics: CHRF-minimized variants and structurally unsound variants for the rejected sonnets. These variants were generated programmatically using prompt templates and then reviewed individually for clarity. We constructed three datasets: a structural data set composed of 432 pairs after cleaning that included prompts with 3,4, and 8 starter lines of a sonnet with the original sonnet as the gold standard and a structurally unsound variant as the non desirable sonnet. We then also made a combined structural and CHRF-minimized data set with 387 preference pairs that foxed on lexical divergence in addition to structural unsoundness.

DPO was implemented with HuggingFace’s datasets and TRL libraries. Preference pairs were loaded with datasets and used to train models through the DPO training framework provided by TRL. DPO

optimizes the following objective for trainable policy π_θ , reference model π_{ref} , prompt x , winning response y_w , losing response y_l , and hyperparameter β [14]:

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

5 Experiments

5.1 Data

Experiments were conducted on two datasets corresponding to our downstream tasks: the Quora Question Pairs dataset for paraphrase detection and a corpus of Shakespeare’s sonnets for sonnet generation. Dataset statistics are summarized in Table 1.

Dataset	Task	Train size	Dev size	Test size
Quora	Paraphrase Detection	141506	20215	40431
Sonnet	Sonnet Generation	143	0	12

Table 1: Dataset Breakdown

5.2 Evaluation method

We evaluated our model using two metrics. For paraphrase detection, we use accuracy, and for sonnet generation, we use the CHRF metric proposed by Popovic[15]. For QLoRA, number of trainable parameters and time per epoch are also evaluated.

5.3 Experimental details

GPT-2 features the following tunable hyperparameters: epochs (default 10), batch size (default 8), learning rate γ (default 1e-5), temperature t (default 1.2), top p (default 0.9), and model size (default gpt2). Both QLoRA and DPO introduce tunable hyperparameters that we manipulate to maximize model performance on our two downstream tasks.

For paraphrase detection with QLoRA, we use the following in addition to GPT’s hyperparameters: LoRA rank r (default 4), quantization q (default 4bit), and presence of last head (no default). We focused on tuning the learning rate γ , LoRA rank r , quantization q (4bit and 8bit), and presence of the last head.

For sonnet generation with DPO, we used the following hyperparameters: DPO batch size (default 8), DPO learning rate γ_{dpo} (default 5e-7), DPO epochs (default 2), and DPO beta β (default 0.1). We focused on tuning the learning rate γ , number of epochs, temperature t , DPO learning rate γ_{dpo} , and DPO beta β hyperparameters. We held all other hyperparameters constant.

All epoch times were obtained on a machine equipped with an NVIDIA RTX 3060 (12GB VRAM), 64GB of system memory, and an AMD Ryzen 7 5800X.

5.4 Results

Model	Last Head	γ	q	r	Parameters	Epoch Time	Dev Acc
gpt2	no	1e-5	n/a	n/a	117M	75 min	0.899*
gpt2	yes	1e-4	4bit	4	1.20M	40 min	0.857
gpt2	yes	1e-5	4bit	4	1.20M	40 min	0.860
gpt2-medium	no	1e-5	n/a	n/a	345M	200 min	0.906
gpt2-medium	yes	1e-5	n/a	n/a	346.63M	210 min	0.907
gpt2-large	yes	1e-4	4bit	4	2.38M	210 min	0.899

Table 2: Paraphrase detection model configuration and results. *: Baseline score.

QLoRA substantially reduces the time per epoch, enabling GPT-2 Large to be trained in approximately the same time required to FFT GPT-2 Medium. While this result initially appears counterintuitive given the difference in parameter counts, GPT-2 Large contains significantly more parameters, and storing these weights, even when quantized, may still incur a large memory overhead. We also observe that adding a final linear classification head improves the performance of GPT-2 Medium on the paraphrase detection task. For GPT-2 models trained with QLoRA, learning rates in the range of 10^{-4} to 10^{-5} appear to perform best, although QLoRA models perform slightly worse overall than FFT GPT-2.

Model	DPO?	γ	β	t	dev CHRF
gpt2	No	1e-3	n/a	1.2	41.525*
gpt2	Yes	1e-3	0.1	1.2	41.701
gpt2	Yes	1e-3	0.01	1.2	41.898
gpt2	Yes	1e-3	0.03	1.2	42.058
gpt2	Yes	1e-3	0.3	1.2	42.057
gpt2	Yes	1e-3	0.3	1.0	42.182

Table 3: Sonnet generation model configuration and results. *: Baseline score.

Prior to implementing DPO, we tuned the learning rate hyperparameter holding all remaining hyperparameters constant at their default values. As learning rate helps balance speed with accuracy, we experimented with larger values for faster convergence. We found $\gamma = 1e - 3$ to be the best rate which makes sense because we are only training the last head which had not been trained prior. This yielded our dev set baseline score of CHRF = 41.525

With the DPO extension, performance increased with the very first iteration and all hyperparameters held at their default values, yielding a dev set score of CHRF = 41.701. This aligns with expected behavior as DPO directly injects human preference data into the model. From here, we hyperparameter tuned the DPO learning rate γ_{dpo} , DPO beta β , and temperature t hyperparameters. While the default DPO learning rate $\gamma_{\text{dpo}} = 5e - 7$ proved to perform the best, we found that $\beta = 0.3$ and $t = 1.0$ performed better than their default values and yielded a dev set score of CHRF = 42.182 on the dev set. We used this configuration to evaluate model performance on the test set as well, yielding a test set score of CHRF = 42.307.

Task	Configuration	Test Score (Accuracy/CHRF)
Paraphrase Detection	Model Size = gpt2, No Last Head	0.898
Sonnet Generation	$t = 1.0, \gamma = 1e - 3, \beta = 0.3$	42.307

Table 4: Best runs with configurations (unlisted hyperparameters take default values)

6 Analysis

6.1 Qualitative

In addition to the quantitative metrics, we consider several examples from the development set to better understand our model’s performance. Overall, for paraphrase detection, our model tends to perform well on paraphrases that occur based on minor lexical variations or rephrasings. However, it seems to struggle when important semantic differences are masked by a larger lexical similarity.

For example, consider the question pair "How do warm and cold fronts form?" and "How does a cold front form?". Our model incorrectly predicts this pair as a paraphrase despite the first question additionally asking for the formation of an entirely different. This example, along with other similar examples, indicate that because of the high lexical similarity, as the first question only has 2 different words, even though those words change semantic meaning, the high degree of similarity obfuscates the change in meaning leading to the incorrect classification. This means that our model might rely too heavily on surface level similarities when determining paraphrases as opposed to considering a deeper semantic meaning.

Conversely, the model sometimes fails to recognize valid paraphrases when the lexical similarity is lower. Specifically, our model incorrectly predicted the pair of questions "Which historical figure do you admire? Why?" and "What historical figure do you respect the most?" to not be paraphrases. This corroborates the idea that our model relies on lexical similarity to classify paraphrases because these questions, despite having the same meaning, use different words in different orders decreasing lexical similarity and ultimately confusing the model. Thus, while our model performs well for paraphrases with subtle lexical differences, it struggle with subtle semantic differences and more pronounced lexical differences.

After adaptation, full fine-tuning, and the DPO extension, our model successfully improved from a dev set baseline performance of 41.525 to 42.182 and surpassed test set baseline performance of 41.208 with a CHRF of 42.307.

Observing the model’s output, it is clear that the model struggled with poem length, rhyme scheme, cohesion, and completion.

Upon achieving dev set baseline performance, all of the generated sonnets were still shorter than the required 14 lines, largely missing the appropriate rhyme scheme, and struggling with cohesion and interpretability. We also observed many examples of nonsense words throughout the sonnets that appeared to be attempts to mimic 13th Century English. See the example here:

Baseline:

If thy soul check thee that I come so near,
Swear to thy blind soul that I was thy will,
And will, thy soul knows, is admitted there.
Make life does most worse all always upon this lord’s lips
From bad mistress agilst as my muse. And it why born no life despise sauous evening,

Even after surpassing baseline performance, many of our sonnets faced the same challenges with completion, rhyme scheme, interpretability and cohesion, but we also observed a more serious loss in line structure. Many lines would collapse into long strings of text, which is likely why we observed improvement in CHRF

Final:

If thy soul check thee that I come so near,
Swear to thy blind soul that I was thy will,
And will, thy soul knows, is admitted there.
O cause thereibe- for which self’s thatst of lies yet gold or in give it it.— even the suffer. behold sc three shea?
land. own treasure six sh that may thy it the you those the end not you thy and nothing foul is maintain sorrow
him would bear thou is mine wisdom shall for self forg be came of the ’ on,’ an female to for the heart to must
true fair thou az be son you it hers, loves time day thee add cr touching if so their sounds to air gone’s come
were sicknessal of fears is that marriage known in heat thy hide to it dear thy wisdom him

7 Conclusion

As expected, QLoRA performs slightly worse than full fine-tuning, but offers the advantage of enabling training on substantially larger models under limited hardware constraints. In our experiments, QLoRA performance appeared relatively invariant to learning rate within the tested range. Notably, this approach allowed us to fine-tune a 762M-parameter model on a single consumer-grade GPU, demonstrating the practical accessibility of parameter-efficient methods. Given additional time, we would conduct further ablation studies on QLoRA, including a direct comparison of 8-bit and 4-bit quantization schemes. We would also investigate the effect of applying LoRA adapters to multilayer perceptron (MLP) layers in addition to attention projections. Finally, with greater computational resources, we would explore scaling to larger models such as GPT-2 XL.

For sonnet generation, DPO interestingly did not have as large of an effect as expected. Part of this can be explained because the initial gold standard we were pushing our model towards was based on the structural soundness of the sonnet through rhyme scheme and maintaining iambic pentameter as opposed to characteristics like correct lexical content like morphology or spelling that CHRF prioritizes.

Ultimately, our experiments demonstrate the potential of combining parameter-efficient techniques like QLoRA with alignment methods like DPO can be promising by making fine-tuning more accessible and more specific to the problem we are trying to solve.

Team contributions (Required for multi-person team)

Mahathi worked on baseline and fine-tuning paraphrase detection, and generating preference pairs for DPO. Tyler worked on the base gpt2 module (and sentiment analysis), quantization, and further fine-tuning both tasks. Taylor worked on the baseline sonnet generation and further optimization with DPO. All members contributed to the paper equally.

References

- [1] Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. Training a helpful and harmless assistant with reinforcement learning from human feedback, 2022.
- [2] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.
- [3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.
- [4] 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.
- [5] Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. Parameter-efficient fine-tuning for large models: A comprehensive survey. *Transactions on Machine Learning Research*, 2024.
- [6] 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. In *International Conference on Learning Representations*, 2022.
- [7] Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 1950–1965. Curran Associates, Inc., 2022.
- [8] Menglin Yang, Jialin Chen, Jinkai Tao, Yifei Zhang, Jiahong Liu, Jiasheng Zhang, Qiyao Ma, Harshit Verma, Regina Zhang, Min Zhou, et al. Low-rank adaptation for foundation models: A comprehensive review. *arXiv preprint arXiv:2501.00365*, 2024.
- [9] Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Gpt3.int8(): 8-bit matrix multiplication for transformers at scale. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 30318–30332. Curran Associates, Inc., 2022.
- [10] Tim Dettmers and Luke Zettlemoyer. The case for 4-bit precision: k-bit inference scaling laws. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 7750–7774. PMLR, 23–29 Jul 2023.

- [11] Elias Frantar, Saleh Ashkboos, Torsten Hoefer, and Dan Alistarh. OPTQ: Accurate quantization for generative pre-trained transformers. In *The Eleventh International Conference on Learning Representations*, 2023.
- [12] Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. SmoothQuant: Accurate and efficient post-training quantization for large language models. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 38087–38099. PMLR, 23–29 Jul 2023.
- [13] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 10088–10115. Curran Associates, Inc., 2023.
- [14] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 53728–53741. Curran Associates, Inc., 2023.
- [15] 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.

A Appendix

Model Size	Parameters	Layers	d_{model}
gpt2	117M	12	768
gpt2-medium	345M	24	1024
gpt2-large	762M	36	1280
gpt2-xl	1542M	48	1600

Table 5: GPT-2 Architectures