Quantized Mixture-of-LoRA-Experts for Low-Cost Training of Large Language Models

A Major Qualifying Project (MQP) Report
Submitted to the Faculty of
WORCESTER POLYTECHNIC INSTITUTE
in partial fulfillment of the requirements
for the Degree of Bachelor of Science in

Computer Science

WPI Authors:

Alpay Ariyak

WPI Project Advisors:

Dr. Xiangnan Kong

Sponsored By:

Together.AI

Date: May 2024

Abstract

This project investigated whether Mixture-of-Experts (MoE) language models could be trained efficiently and effectively using consumer hardware by combining the MoE architecture with low-cost training methods such as Quantized Low-Rank Adaptation (QLoRA). We focused on an approach that involved training multiple QLoRA adapters separately on different data subsets and merging them during inference using various routing strategies. The goal was to empirically assess whether these QLoRA-MoE hybrid models could enable high-quality language models to be developed accessibly without requiring massive compute resources, while still approaching the performance of full fine-tuning. Our experiments demonstrate the exciting potential of QLoRA-MoE, as it can significantly outperform standard QLoRA and achieve results remarkably close to full fine-tuning while requiring substantially less memory and compute. The best QLoRA-MoE configuration used centroid-based gating and a moderate number of experts, suggesting that relatively simple routing strategies can be effective. However, there is still room for further exploration and optimization of the gating mechanisms and expert architectures. While QLoRA-MoE did not consistently outperform full fine-tuning, its ability to come close while using substantially fewer resources is highly promising. We believe this work demonstrates the potential of combining quantization with mixture-of-experts routing for efficient and effective adaptation of large language models, and we hope it will inspire further research in this direction.

Acknowledgements

- Together.AI, for sponsoring this project.
- Dr. Xiangnan Kong, for advising this project on my end.
- Artem Yatsenko, for providing his documentation on our development of this project from last year. [13]

Core Team and Contributors

- **Core Team**
  - Farouk El
  - Prateek Yadav
  - Artem Yatsenko
  - Alpay Ariyak
  - Harrison Kinsley
- **Contributors**
  - Yam Peleg
  - Alignment Lab
  - Teknium
  - interstellarninja
  - Jeremy Howard
  - Yaroslav
  - Caseus
  - Premeditator
  - Luigi D.
  - Kranthi
  - and many others

Contents

1 Introduction

Large language models have achieved impressive results on a wide range of natural language tasks, but their training is extremely computationally expensive, often requiring massive clusters of GPUs or TPUs. Mixture-of-Experts (MoE) architectures aim to decouple model size from inference speed by conditionally activating different subsets of parameters for each input. This project explored whether combining MoE with parameter-efficient methods like Low-Rank Adaptation (LoRA), specifically, Quantized LoRA (QLoRA), could enable high-quality models to be trained more efficiently on low-cost consumer hardware.

Code and Data

The code for this project is available on GitHub at the following repository:
https://github.com/SkunkworksAI/hydra-moe.
Our datasets and models are available at the following HuggingFace Organization Repository:
https://huggingface.co/HydraLM.

1.1 Background

Mixture-of-Experts (MoE) is a technique for scaling up machine learning models by combining multiple specialized submodels or "experts". In an MoE architecture, a gating function is used to route each input to the most relevant experts, and the outputs of the experts are then combined to produce the final prediction. This allows the model to leverage specialized knowledge for different types of inputs, while also being computationally efficient by only activating a subset of the experts for each example. However, training MoE language models can still be prohibitively expensive, as it requires a lot of GPU memory.

Another line of work has focused on developing parameter-efficient methods for fine-tuning large language models, such as adapters [7], prompt tuning [9], and Low-Rank Adaptation (LoRA) [8]. These methods aim to adapt pre-trained models to new tasks by learning a small number of additional parameters while keeping the original model weights frozen. This can significantly reduce the computational cost of fine-tuning, making it more feasible to train large models on limited hardware.

LoRA and QLoRA[1] have emerged as the main approaches for efficient fine-tuning. LoRA It works by learning a low-rank decomposition of the weight matrices in the model, which allows for adapting the model to new tasks with minimal additional parameters. LoRA has been shown to achieve performance comparable

to full fine-tuning on a wide range of natural language tasks while being much more computationally efficient. It's important to note that while we refer to our overall approach as QLoRA-MoE, the "Q" (quantization) only applies to the base model, not the adapters themselves. The adapters are trained using the standard LoRA technique, while the base model is quantized to 4 bits to further reduce memory usage. Therefore, QLoRA refers to the combination of a 4-bit quantized base model with LoRA adapters, rather than quantized adapters.

2 Related Work

There has been significant prior research on both Mixture-of-Experts models and parameter-efficient fine-tuning methods. Some of the most relevant works that informed our approach include:

- GLaM: Efficient Scaling of Language Models with Mixture-of-Experts[2]
- DEMix Layers: Disentangling Domains for Modular Language Modeling[5]
- Branch-Train-Merge: Embarrassingly Parallel Training of Expert Language Models[10]
- Scaling Expert Language Models with Unsupervised Domain Discovery[6]
- Soft Merging of Experts with Adaptive Routing[11]
- Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity[3]
- AdapterFusion: Non-Destructive Task Composition for Transfer Learning[12]

Our work aims to combine insights from these papers by exploring QLoRA as a more parameter-efficient way to implement MoE models.

2.1 Project Objectives

2.1.1 Evaluate QLoRA-based MoE approach

Our main objective is to assess the effectiveness of the proposed QLoRA-MoE approach for training large language models on low-cost hardware. This involves empirically comparing the performance of models trained using our method to standard QLoRA and LoRA baselines, as well as full fine-tuning when feasible. We aim to understand the trade-offs between model quality, training efficiency, and inference latency, and identify the most promising configurations for practical use.

2.1.2 Assessing feasibility as low-cost training approach

A key goal of this work is to determine whether the QLoRA-MoE approach can enable training high-quality language models on consumer GPUs, making it more accessible to a wider range of researchers and practitioners. We carefully track the computational requirements of our experiments and aim to provide insights and recommendations to guide future work on democratizing large language model development.

3 QLoRA-based Mixture-of-Experts

Our approach to training large language models on modest hardware involves combining the ideas of Mixture-of-Experts (MoE) and Quantized Low-Rank Adaptation (QLoRA). However, we differ from the standard MoE language model architecture in several key ways.

Instead of starting with an MoE model, we begin with a dense pre-trained model (in this case, LLaMA-2) and aim to convert it into an MoE-like structure through external routing. Specifically, we train with quantized base model multiple LoRA modules to serve as "experts" that specialize in different subsets of the data. LoRA is a parameter-efficient fine-tuning technique that learns a small number of adaptation matrices for each model layer while keeping the original weights frozen. It can achieve comparable performance to full fine-tuning at a fraction of the computational cost. What we are doing is QLoRA, the only difference being that the base model is quantized during training.

During inference, we use a routing function to assign each incoming query to the relevant LoRA adapter experts based on the semantic content of the prompt. The adapter weights from these experts are then combined (e.g. via weighted averaging) to produce the final prediction. This differs from the typical MoE approach of routing individual tokens and selectively activating experts at each layer. By applying routing at the prompt level and using LoRA modules as experts, we can increase efficiency, as the inference operation would only be with a single adapter since we're merging them.

The key insight is that using LoRA adapters provides a way to train multiple specialized adapters that can be dynamically combined at inference time, similar in spirit to an MoE model but without the need for a custom sparse architecture. This allows us to increase the effective capacity of the model while keeping training and inference costs manageable. Our hypothesis is that this QLoRA-based MoE approach can yield a powerful and efficient model that can be trained on modest hardware by leveraging the combined benefits of adaptive routing and parameter-efficient fine-tuning.

3.1 Quantized Training of the LoRA Experts

The first step in our approach is to train a set of LoRA adapter modules that specialize in different subsets of the training data. To obtain these subsets, we employ unsupervised clustering techniques to partition a large, diverse corpus into more focused clusters. Specifically, we use k-means clustering based on sentence embeddings obtained from a pre-trained language model (in our experiments, E5-Large). This allows us to group semantically similar examples together without relying on predefined labels or categories.

The motivation for this clustering step is twofold. First, it allows each LoRA expert to specialize in a particular domain or topic, which may lead to improved performance on queries related to that area. Second, by breaking up the training data into smaller, more coherent subsets, we can more effectively leverage the available examples without running into the model capacity bottleneck as quickly. This is especially important when using parameter-efficient methods, such as QLoRA and LoRA, which have limited ability to absorb large, diverse datasets.

Once the data is partitioned into k clusters, we train a separate LoRA adapter on each cluster. We keep the base pre-trained model frozen and only update the LoRA adapter parameters. This process yields a collection of k specialized adapters that can be combined at inference time using a routing strategy.

3.2 Prompt-based Routing

To utilize the specialized knowledge captured by the LoRA experts, we need a way to determine which adapters are most relevant for a given input prompt. We experimented with several routing strategies:

1. **Similarity-based routing**: Compute the cosine similarity between the input prompt embedding and the centroid embedding of each cluster (obtained during the k-means clustering step). Then, select the top m most similar clusters and use their corresponding LoRA adapters for inference. The weights from these adapters are merged using a weighted average, where the weights are proportional to the similarity scores.

2. **Learned routing**: Train a supervised classifier (e.g. BERT) to predict the most relevant cluster(s) for a given prompt. This is done by using the cluster assignments from the k-means step as labels and fine-tuning the classifier on a held-out portion of the data. At inference time, the prompt is first passed through the router to select the top m adapters, whose weights are once again weight-averaged to produce the final prediction.

3. **Hybrid routing**: Combine the similarity-based and learned routing approaches by using a

weighted combination of the two scores (cosine similarity and classifier probability) to rank the adapters. The weights can be tuned as hyperparameters or learned end-to-end.

To make the routing process more concrete, let's walk through an example. Suppose we have a QLoRA-MoE model with $k = 32$ adapters and we receive the following query that is semantically a combination of nutrition/food, calculation, and coding:

> *Write a Python program that takes in the macros ("protein", "calories", "fat", and "carbohydrates") of food by 100g, the weight of the food, and returns the macros.*

First, we embed the query using the same pre-trained model that was used for clustering (E5-Large in our case). Then, we compute the cosine similarity between this embedding and the 32 cluster centroids. Suppose the top 3 most similar clusters are:

- Cluster 21 (nutrition/cooking): 0.8 similarity
- Cluster 13 (programming): 0.6 similarity
- Cluster 6 (biology): 0.2 similarity

Next, we pass the query through our learned router model (e.g. E5 fine-tuned on the cluster labels). The router assigns the following probabilities to the top 3 clusters:

- Cluster 21: 0.7
- Cluster 13: 0.25
- Cluster 6: 0.05

Taking a weighted average of the similarity scores and router probabilities (with equal weight), we arrive at the following ranking:

- Cluster 21: $0.5 * (0.8 + 0.7) = 0.75$
- Cluster 13: $0.5 * (0.6 + 0.25) = 0.425$
- Cluster 6: $0.5 * (0.2 + 0.05) = 0.125$

Based on this, we merge the 3 LoRA adapters corresponding to clusters 21, 13, and 6 (assuming $m = 3$) with weighted averaging to produce the final output.

This example illustrates how our QLoRA-MoE approach can dynamically combine knowledge from different experts based on the semantics of the input prompt. By leveraging the specialized adapters trained on focused data subsets, we aim to improve the model's performance on a wide range of tasks while keeping the computational cost manageable.

However, there are some limitations and open questions with this approach. First, the router is trained to predict the most semantically similar clusters, but this may not always correspond to the optimal combination of adapters for generating a good response. There could be cases where less similar but more informative experts are neglected. Second, the weighted average is a relatively simple way to combine adapter weights, and may not capture more complex interactions between experts. More sophisticated ensembling methods could be explored.

Despite these challenges, our initial experiments show promising results compared to standard unquantized LoRA fine-tuning and even the full fine-tune. By allowing the model to adapt and specialize in a more fine-grained way, we can potentially unlock better performance without sacrificing efficiency. In the next section, we describe our experiments and findings in more detail.

3.3 Results

To evaluate the quantized multi-LoRA adapter approach, we measured performance on the LM Evaluation Harness [4], a suite of diverse language understanding tasks. We compare our 2 trained QLoRA-MoEs, one trained with 32 clusters and the other with 16, to baseline LoRA, QLoRA, and full-finetunes on the same corpus without any clustering.

Title	Gating	LoRA Alpha	# of Experts	Total	ARC_C	ARC_E	BoolQ	HellaSwag	OpenBookQA	Piqa	Winogrande
Full Fine-Tune				68.57%	0.4974	0.7614	0.7997	0.778	0.466	0.7987	0.6985
QLoRA MoE-16	Centroid	1	16	68.26%	0.5	0.7437	0.804	0.7733	0.458	0.7949	0.704
QLoRA MoE-16	Centroid	2	16	68.24%	0.4991	0.7424	0.8119	0.7714	0.454	0.7911	0.7072
QLoRA MoE-32	Transformer	8	32	68.05%	0.4923	0.7529	0.815	0.7687	0.446	0.7867	0.7017
LoRA		16		67.75%	0.4872	0.7487	0.7979	0.7629	0.46	0.7889	0.6969
QLoRA MoE-32	Transformer	16	32	65.97%	0.4659	0.7336	0.8067	0.7495	0.408	0.7764	0.678
QLoRA MoE-32	Combined	16	32	65.58%	0.4599	0.7285	0.8107	0.7408	0.41	0.7731	0.6677
QLoRA		16		63.32%	0.4215	0.6587	0.8006	0.7391	0.412	0.7388	0.6614
QLoRA MoE-32	Centroid	16	32	62.78%	0.4317	0.6974	0.7911	0.6947	0.376	0.7704	0.633
QLoRA MoE-16	Centroid	16	16	61.05%	0.4113	0.6684	0.8	0.663	0.39	0.747	0.5935

Figure 1: QLoRA-MoE vs Full Fine-Tuning, QLoRA, LoRA

Across most tasks, we found that using multiple LoRA adapters, even with our quantized base model, outperformed the single LoRA baseline, and almost all configurations of our MoEs beat the single quantized LoRA(QLoRA) baseline.

Our approach also came surprisingly close to the performance of full fine-tuning.

In terms of the routing strategies, we found that cosine similarity was generally the most robust, while the learned classifier had the potential to outperform it but was more sensitive to hyperparameters and required more tuning.

4 Conclusion

In this project from 2023, we proposed QLoRA-MoE, a hybrid approach that combines 4-bit quantized LoRA with mixture-of-experts routing, aiming to enable efficient and effective fine-tuning of large language models on resource-constrained hardware. The best QLoRA-MoE configuration used centroid-based gating and a moderate number of experts, suggesting that relatively simple routing strategies can be effective. However, there is still room for further exploration and optimization of the gating mechanisms and expert architectures.

Our experimental results show that QLoRA-MoE not only surpasses standard QLoRA in performance but also seemingly approaches full fine-tuning while utilizing less memory and computational resources. This may be very case-specific to the exact conditions we did our data collection in, but this can indicate the possibility of the approach being feasible as a very effective low-cost option with low accuracy loss.

There have been many developments since we worked on this project, and the results need to be tested further, but overall we believe this work highlights the valuable potential of combining quantization with mixture-of-experts for efficient and effective adaptation of large language models.

References

[1] Tim Dettmers et al. *QLoRA: Efficient Finetuning of Quantized LLMs*. 2023. arXiv: 2305.14314 [cs.LG].

[2] Nan Du et al. *GLaM: Efficient Scaling of Language Models with Mixture-of-Experts*. 2022. arXiv: 2112.06905 [cs.CL].

[3] William Fedus, Barret Zoph, and Noam Shazeer. *Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity*. 2022. arXiv: 2101.03961 [cs.LG].

[4] Leo Gao et al. *A framework for few-shot language model evaluation*. Version v0.4.0. Dec. 2023. DOI: 10.5281/zenodo.10256836. URL: https://zenodo.org/records/10256836.

[5] Suchin Gururangan et al. *DEMix Layers: Disentangling Domains for Modular Language Modeling*. 2021. arXiv: 2108.05036 [cs.CL].

[6] Suchin Gururangan et al. *Scaling Expert Language Models with Unsupervised Domain Discovery*. 2023. arXiv: 2303.14177 [cs.CL].

[7] Neil Houlsby et al. *Parameter-Efficient Transfer Learning for NLP*. 2019. arXiv: 1902.00751 [cs.LG].

[8] Edward J. Hu et al. *LoRA: Low-Rank Adaptation of Large Language Models*. 2021. arXiv: 2106.09685 [cs.CL].

[9] Brian Lester, Rami Al-Rfou, and Noah Constant. *The Power of Scale for Parameter-Efficient Prompt Tuning*. 2021. arXiv: 2104.08691 [cs.CL].

[10] Margaret Li et al. *Branch-Train-Merge: Embarrassingly Parallel Training of Expert Language Models*. 2022. arXiv: 2208.03306 [cs.CL].

[11] Mohammed Muqeeth, Haokun Liu, and Colin Raffel. *Soft Merging of Experts with Adaptive Routing*. 2023. arXiv: 2306.03745 [cs.LG].

[12] Jonas Pfeiffer et al. *AdapterFusion: Non-Destructive Task Composition for Transfer Learning*. 2021. arXiv: 2005.00247 [cs.CL].

[13] Artem Yatsenko. *QLora-MoE: What Worked and What Didn't*. Accessed: 2024-05-05. 2023. URL: https://sumo43.github.io/jekyll/update/2023/08/25/lora_moe.html.