Fasting NLP:Slimming Down Models with QLoRA

Stanford CS224N {Custom, Default} Project

Ricardo Carrillo
SCPD
Stanford University
racr@stanford.edu

Abstract

Abstract

The field of Natural Language Processing (NLP) has seen rapid advancements, with models achieving exceptional performance in tasks such as sentiment analysis, paraphrase detection, and semantic similarity. However, these state-of-the-art models often contain hundreds of millions of parameters, making adaptation to new tasks computationally challenging and prone to overfitting. Finetuning Large Language Models (LLMs) like the LLaMA model, with 65 billion parameters, is resource-intensive, requiring over 780 GB of GPU memory for standard 16-bit finetuning. The QLoRA (Quantized Low-Rank Adapter) method introduces a novel approach by demonstrating the feasibility of finetuning a quantized 4-bit model without performance degradation. QLoRA employs high-precision quantization to reduce a pretrained model to 4-bit and incorporates a small set of learnable Low-rank Adapter (LoRA) weights, enhancing efficiency and cost-effectiveness while preserving performance. Our adaptation of QLoRA utilized the `to_NF4` function to quantize the linear layer applied to LoRA, applied to all linear layers except those involved in specific NLP tasks. We also integrated extensions such as annealed sampling, Sentence Concat for Paraphrase detection and Semantic Textual Similarity, and SST Pre-Training with MASKED LM. Despite not employing double quantization, our method demonstrated that efficient quantization and dequantization functions are critical for computational speed. Our GPU memory usage was significantly reduced from 3760 MB to 1783 MB, highlighting the efficiency of our approach. Experimental results showed improvements: sentiment analysis accuracy on the SST dev set to 0.506, paraphrase detection to 0.9, and semantic textual similarity to 0.882. These findings suggest that sophisticated regularization methods could further enhance model performance, especially for tasks like Paraphrase Detection and Semantic Textual Similarity.

1 Key Information to include

- TA mentor: Neil Nie, External collaborators (No), External mentor (No), Sharing project (No)

2 Introduction

The field of Natural Language Processing (NLP) has witnessed rapid advancements in recent years, with models achieving unprecedented performance in a variety of tasks such as sentiment analysis, paraphrase detection, and semantic similarity. However, these state-of-the-art models often encompass hundreds of millions of trainable parameters, making their adaptation to new tasks computationally

challenging (Maziarka and Danel, 2021). Furthermore, these models are prone to overfitting and struggle with data scarcity issues (Chen et al., 2024) (Chen et al., 2021). Multitask Learning (MTL) has been proposed as an effective strategy to enhance language representations, improve model generalization, and optimize resource utilization to address these challenges. Among the various models employed for multitask learning in NLP, BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2019) has emerged as a leading choice due to its robust performance across a wide range of tasks. A plethora of methodologies have been explored to tackle the challenges associated with large models, leading to substantial improvements in BERT-based multitask models. Finetuning Large Language Models (LLMs) is a potent strategy to augment their performance and customize their behaviors to cater to specific tasks. However, the finetuning process for extremely large models, such as the LLaMA model with 65 billion parameters, is exceptionally resource-intensive, requiring over 780 GB of GPU memory for standard 16-bit finetuning. While recent quantization methods have successfully reduced the memory footprint of LLMs, these techniques are typically confined to inference and fail during training. The QLoRA (Dettmers et al., 2023) paper introduce, a novel method that overcomes this limitation by demonstrating the feasibility of finetuning a quantized 4-bit model without any performance degradation for the first time. QLORA employs a high-precision quantization technique to reduce a pretrained model to 4-bit, and subsequently incorporates a small set of learnable Low-rank Adapter (LoRA) weights. This innovative approach not only enhances the efficiency and cost-effectiveness of finetuning but also preserves the high performance of large language models. As a result, QLORA opens up new avenues for more practical and scalable applications in NLP. This paper presents the design, implementation, and evaluation of QLORA, providing a comprehensive understanding of its potential and implications for the NLP community. This new model, together with the challenge of the magnitude of the most recent LLM's and their fine-tuning motivate us to explore the QLoRA method in this project as well as some tools for tuning models with imbalance in their datasets.

3 Related Work

The Bidirectional Encoder Representations from Transformers (BERT) model, introduced by (Devlin et al., 2019), has achieved state-of-the-art performance in numerous Natural Language Processing (NLP) tasks. This model significantly mitigates the requirement for labeled data by pre-training on unlabeled data across various tasks. Initially, BERT is trained on plain text for masked word prediction and next sentence prediction tasks, a phase referred to as pretraining. Subsequently, it is fine-tuned on a specific linguistic task with additional task-specific layers using task-specific training data.

In 2019, Liu et al. proposed a multitask learning framework for BERT that concurrently learns to classify multiple text attributes, such as pairwise text classification and text similarity. The lower layers, which are the text encoding layers, are shared across all tasks, while the top layers are task-specific, integrating different types of Natural Language Understanding (NLU) tasks.

QLORA, on the other hand, achieves high-fidelity 4-bit fine-tuning through two proposed techniques—4-bit NormalFloat (NF4) quantization and Double Quantization. QLORA utilizes one low-precision storage data type, typically 4-bit, and one computation data type, usually BFloat16. In practical terms, this implies that whenever a QLORA weight tensor is utilized, the tensor is dequantized to BFloat16, followed by a matrix multiplication in 16-bit.

Furthermore, (Stickland and Murray, 2019) proposed an innovative method for scheduling training. Initially, the tasks are sampled proportionally to their training set size. However, to avoid interferences, the weighting is reduced to enable tasks to be sampled more uniformly. This approach ensures a balanced representation of all tasks during the training process.

4 Approach

4.1 Annealed Sampling. In our evaluation of the base model, we observed significant disparities in accuracy across different subtasks, we achieved SST Dev: 0.302, Para Dev: 0.696 and STS Dev: 0.272. Paraphrase detection achieved the highest accuracy, followed by semantic textual similarity, with sentiment classification exhibiting the lowest performance. These discrepancies can be attributed to the varying amounts of training data available for each task: 141,498 examples for paraphrase detection, 8,544 for sentiment analysis, and 6,040 for semantic textual similarity. This finding underscores a limitation of the base model's round-robin training approach. As noted by (Stickland and Murray, 2019), this method, which samples batches from each dataset in a predetermined sequence, may deplete smaller datasets prematurely, potentially hindering the training process. Annealed sampling presents a viable solution by selecting tasks based on the dataset size, modeled as:

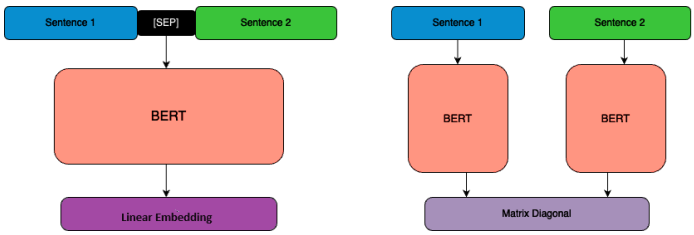
$$p_i \propto N_i^{\alpha}$$

where p_i represents the probability of selecting a batch from task i, and N_i denotes the number of training examples in task i. We adopt (Stickland and Murray, 2019) definition for α:

$$\alpha = 1 - 0.8\frac{e-1}{E-1}$$

where e is the current epoch and E is the total number of epochs. This scheme prioritizes training tasks with larger datasets at the onset of training. As training progresses, the probability of selecting each task becomes more uniform, ensuring that no task disproportionately dominates the training process. This approach mitigates the risks of overfitting and underfitting, thereby enhancing the overall model performance.

4.2.- Sentence Concat for Paraphrase detection and Semantic Textual Similarity In the realm of Natural Language Processing, tasks such as Semantic Textual Similarity and Paraphrase Detection necessitate the evaluation of a pair of input sequences to yield a prediction. This presents a formidable challenge for models like BERT, which are traditionally architected to process single input sequences. Originally, we took the diagonal of the embedding multiplication as the similarity between embeddings, with very low than expected results. To surmount this obstacle, we use the bi-encoding strategy, as delineated by (Reimers and Gurevych, 2019), which independently processes each input sequence via BERT. Subsequently, the embeddings derived from each sequence are amalgamated to generate a prediction pertinent to the sentence pair. The method of combination is contingent on the task at hand: for paraphrase detection, the embeddings are concatenated and a linear classifier is employed, while for semantic textual similarity, the cosine similarity between the two embeddings is utilized as a measure of resemblance.



4.3 SST Pre-Training with MASKED LM After several iterations, we observed that the SST accuracy remained around 0.5. To address this, we included a pre-training phase using the same dataset before commencing the multi-task training. This pre-training was conducted using MASKED LM, as suggested by Devlin et al. Specifically, the method follows:

> The training data generator chooses 15% of the token positions at random for prediction. If the i-th token is chosen, we replace the i-th token with (1) the [MASK] token 80% of the time, (2) a random token 10% of the

time, (3) the unchanged i-th token 10% of the time. Then, T_i will be used to predict the original token with cross entropy loss.

After implementing this method for approximately 20 epochs, the multi-task training began. Despite increasing the training speed, this effort did not achieve the expected improvement in accuracy performance.

4.4 QLoRA(Quantized Low-Rank Adapter) is an innovative approach to efficiently finetune large language models (LLMs) while significantly reducing memory usage. This method enables the finetuning of models with up to 65 billion parameters on a single 48GB GPU, maintaining the performance of full 16-bit finetuning.

QLoRA introduces several key innovations:

1. **4-bit NormalFloat (NF4) Quantization**: This data type is optimized for normally distributed weights, ensuring equal value distribution across quantization bins. NF4 resizes the input tensor to fit a fixed range, preserving the precision of the quantized data.
2. **Double Quantization**: This technique reduces memory usage by quantizing the quantization constants. It applies a second layer of quantization, lowering the bit requirement per parameter and saving memory.
3. **Paged Optimizers**: These optimizers handle memory spikes during gradient checkpointing using NVIDIA's unified memory feature, which automatically transfers memory between CPU and GPU. This prevents out-of-memory errors and allows efficient processing of large models on single GPUs.

QLoRA operates by backpropagating gradients through a frozen, 4-bit quantized pretrained model into Low-Rank Adapters (LoRA). LoRA adapters are small sets of trainable parameters that enhance the fixed pretrained model weights. During finetuning, gradients pass through the fixed model weights to the adapters, which are updated to optimize the loss function, maintaining high performance with reduced memory overhead.

Our adaptation of QLoRA from scratch does not include all the functionalities described in the original paper. We used the `to_NF4` function to QLorize the linear layer applied to LoRA, whose weights had to be quantized in the forward method. The QLoRA paper suggests applying the method to all linear layers: "we find that the most critical LoRA hyperparameter is how many LoRA adapters are used in total and that LoRA on all linear transformer block layers," unlike LoRA, which in its paper suggests applying LoRA only to the attention layers. In our case, we applied the method to all linear layers except the ones included for the NLP tasks (SST, PARA, STS).

Although we experimented with several custom functions to quantize and dequantize, we ultimately used methods from the Torchstone library: `to_nf4` for quantization and `linear_nf4` for dequantization. Below, we show our forward method inside the `QLoRALayer` class.

```
def forward(self, x):
    # Call into torchao's linear_nf4 to run linear forward pass w/quantized weight.
    x = x.to(device)
    self.linear_weight = nn.Parameter(self.linear_weight.to(device))
    frozen_out  = linear_nf4(x, self.linear_weight).to(device)
    # lora_a projects inputs down to the much smaller self.rank,
    # then lora_b projects back up to the output dimension
    lora_out = self.lora_A(self.lora_B(self.dropout(x))).to(device)
    # Finally, scale by the alpha parameter (normalized by rank)
    # and add to the original model's outputs
    return frozen_out + (self.alpha / self.rank) * lora_out
```

Even though our method did not double quantize, it took almost twice as long to run as our training without QLoRA, highlighting the importance of efficient quantization and dequantization functions for computational speed. However, our GPU memory usage was reduced from 3760 MB to 1783 MB, as monitored with the `nvidia-smi` command on our VM.

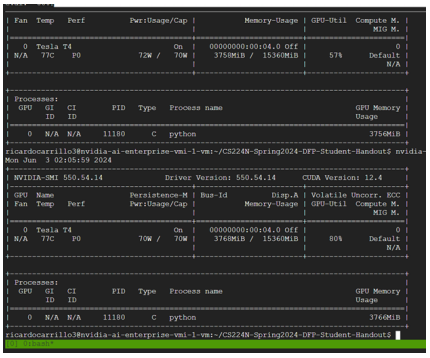


Figure 1: NON-QLORA TRAINING GPU MEMORY

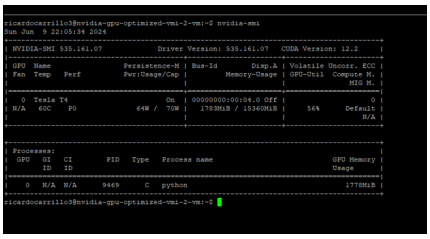


Figure 2: QLORA TRAINING GPU MEMORY

5 Experiments

This section contains the following.

5.1 Data

For the First phase of the project we used Stanford Sentiment Treebank (SST) (Socher et al., 2013) and the CFIMDB dataset for Sentiment Analysis. SST has 215,154 phrases but we will use a subset of 11,855 samples: train (8,544), dev (1,101), and test (2,210). CFIMDB has 2,434 samples: train (1,701), dev (245), and test (488). For the second phase of the project where we develop the extensions, we used The Stanford Sentiment Treebank Dataset (SST) consists of movie reviews (9,645 examples) from Rotten Tomatoes. Specifically, we employ fine-grained sentiment analysis, SST-5, where sentiments range from 0 (negative) - 4 (positive) with train (8,544 examples), dev (1,101 examples), test (2,210 examples),.we have available for Paraphrase Detection the Quora Dataset with 404,298 question pairs: train (283,010), dev (40,429), and test (80,859). For Semantic Textual Similarity (STS), we will use the SemEval STS Benchmark dataset with train (6,040), dev (863), and test (1,725).

5.2 Evaluation method

For the tasks Sentiment Analysis and Paraphrase Detection, accuracy is the evaluation metric evaluation, and for the Semantic Textual Similarity (STS) tasks, we will use the Pearson correlation coefficient.

5.3 Experimental details

Report how you ran your experiments (e.g., model configurations, learning rate, training time, etc.)

TEST	FINE-TUNE	LR	EPOCHS	SST DEV	PARA DEV	STS DEV	DROPOUT
1	FULL MODE	1.00E-05	10	0.302	0.696	0.272	0.3
2	FINE TUNING	1.00E-03	10	0.41	0.629	0.457	0.3
3	FULL MODE	1.00E-05	10	0.5	0.829	0.874	0.3
4	FULL MODE	1.00E-05	20	0.486	0.845	0.882	0.3
5	FULL MODE	1.00E-05	10	0.506	0.9	0.882	0.3
6	FULL MODE	1.00E-05	5	0.498	0.9	0.883	0.4
7	FINE TUNNING	1.00E-03	3	0.393	0.67	0.284	0.3
8	FINE TUNNING	1.00E-03	3	0.379	0.675	0.334	0.3
9	FULL MODE	1.00E-05	10	0.48	0.868	0.864	0.4
10	FULL MODE	1.00E-05	10	0.491	0.9	0.879	0.4

Table 1: Results for 10 Test Part 1/2

TEST	TYPE OF SAMPLING	BATCH	PARA PREDICTION	STS PREDICTION	PRE-TRAIN STS
1	ROUND ROBIND	8	MATRIX DIAGONAL	MATRIX DIAGONAL	NO
2	ANNEALED SAMP.	8	MATRIX DIAGONAL	CONCATENATION	NO
3	ANNEALED SAMP.	8	MATRIZ DIAGONAL	CONCATENATION	NO
4	ANNEALED SAMP.	16	MATRIZ DIAGONAL	CONCATENATION	NO
5	ANNEALED SAMP.	8	CONCATENATION	CONCATENATION	NO
6	ANNEALED SAMP.	8	CONCATENATION	CONCATENATION	NO
7	ANNEALED SAMP.	8	CONCATENATION	CONCATENATION	YES-PARTIAL
8	ANNEALED SAMP.	8	CONCATENATION	CONCATENATION	YES
9	ANNEALED SAMP.	8	CONCATENATION	CONCATENATION	YES
10	ANNEALED SAMP.	8	CONCATENATION	CONCATENATION	YES

Table 2: Resuls for 10 Tests Part 2/2

5.4 Results

Report the quantitative results that you have found. Use a table or plot to compare results and compare against baselines.

- Our results reported in Test and Test Leaderboard are SST test accuracy: 0.503, Paraphrase test accuracy: 0.899, STS test correlation: 0.877. The position in the Dev Leaderboard at the moment is 37th.
- The values are a little lower than expected especially for the SST case, I think that the regularization method could have helped to improve the accuracy. The time factor played a role.
- The topical time for a Full Mode Training was about 10 hours.
- The parameters changes because the New Lora Layers we have :Original parameter count: 109483778, QLoRA parameter count: 120115202, Parameter increase after QLoRA: 9.71

6 Analysis

Analyzing the results shows that there is a lot of room to work on the Analysis Sentiment SST dataset, starting with regularization. The accuracy percentage with respect to the training set could reach 99.99% in the 5th epoch while the accuracy with the dev set was still in the high 80's, implying that regularization was necessary. I thought that adding a higher dropout of 0.3 to 0.4 was like a "brute force" measure for regularization but no, more sophisticated regularization is needed. Also, better regularization could further improve Paraphrase Detection and Semantic Textual Similarity.

7 Conclusion

An important finding was to use methods in the difficulty of handling unbalanced datasets, also of the multiple fine-tuning techniques it is an art to find the techniques with the most impact. A limitation to use QLoRA in all layers including the NLP tasks, was the SCALE that was set to 64, changing it to degrade the accuracy and make it dynamic could be a major commitment for the project time. The QLora method may not have the expected speed impact depending on the quantization library to NF4, future exploration on the creation of libraries to convert data types could be interesting.

8 Ethics Statement

Challenges and possible Mitigations strategies As mitigating strategies, the use of tools such as the one proposed by (Bolukbasi et al., 2016) developing algorithms for "debiasing" training corpora are fundamental. From my point of view promoting the proper use of these models could be propelled from 3 areas of society, academia, government regulatory framework, and private sector. The academia stimulating

research in this area, the government promoting with fiscal benefits the Non BIAS Certification of AI Applications, and the private sector encouraging the creation of an Ecosystem of Companies whose purpose is to certify third-party applications as Non BIAS. The consumers would prefer a Non-BIAS Certified app.

References

Tolga Bolukbasi, Kai-Wei Chang, James Zou, Venkatesh Saligrama, and Adam Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings.

Shijie Chen, Yu Zhang, and Qiang Yang. 2024. Multi-task learning in natural language processing: An overview.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Łukasz Maziarka and Tomasz Danel. 2021. Multitask learning using bert with task-embedded attention. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–6.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Asa Cooper Stickland and Iain Murray. 2019. Bert and pals: Projected attention layers for efficient adaptation in multi-task learning.

A Appendix (optional)

If you wish, you can include an appendix, which should be part of the main PDF, and does not count towards the 6-8 page limit. Appendices can be useful to supply extra details, examples, figures, results, visualizations, etc. that you couldn't fit into the main paper. However, your grader *does not* have to read your appendix, and you should assume that you will be graded based on the content of the main part of your paper only.