DoRA: Decomposed Low-Rank Adaptation

Rolando Rodríguez[1], Ishaan Nanal[1], Chris Fernandes[1], Gerardo Montemayor[1]

[1]Cornell University, Bowers College of Computing and Information Sciences
https://github.com/rolanrod/cs-4782-final-project

1 Introduction and Motivation

Adapting powerful models towards accomplishing specific tasks has become a central challenge in modern machine learning, and as models continue to grow in size and complexity, the need for efficient fine-tuning methods has become increasingly important. Fully fine-tuning a base model, while less intensive than pre-training, still comes at too high a computational cost for many would-be users of large models.

LoRA: Low-Rank Adaptation of Large Language Models introduced a breakthrough in 2021 that circumvented the need for full backpropagation over all parameters in a base model and pioneered parameter-efficient fine-tuning (PEFT) approaches to tackling this problem. Suppose we have a weight $W \in \mathbb{R}^{1024\times1024}$ that we need to update in fine-tuning. This update can be thought of as a matrix addition of our changes with W, namely, $W' = W + \Delta W$. LoRA proposes to represent this update ΔW as a low-rank decomposition: $\Delta W = AB$, where $A \in \mathbb{R}^{1024\times r}$ and $B \in \mathbb{R}^{r\times 1024}$ for a small $r \ll 1024$. Rather than updating the full matrix W during fine-tuning, LoRA freezes W and trains only the much smaller matrices A and B, reducing the number of trainable parameters and the associated memory and compute costs– if we set the hyperparamater r to 10, we can achieve a 98% reduction in parameter count for this example. This approach enables scalable fine-tuning across a variety of downstream tasks without the prohibitive expense of retraining the full model.

LoRA indeed makes a previously computationally infeasible problem much more tractable in practical settings, but it is fundamentally still just an approximation of full fine-tuning and often lags behind it in performance. The key insight behind LoRA—and low-rank approximations more broadly—is that many high-dimensional matrices used in deep learning contain significant redundancy. By removing linearly dependent directions and projecting updates into a lower-dimensional subspace, LoRA assumes that most of the meaningful variation in model updates lies within a low-rank manifold.

However, this compression comes at a cost. The matrix product AB entangles both the *direction* and *magnitude* of the update in a single term. As a result, LoRA's updates may not capture the full representational dynamics of full fine-tuning, particularly in settings where update direction and scaling behave differently across tasks or layers. Below is a plot produced by Liu et. al that compares how full fine-tuning (FT) and LoRA are able to attend to changes in magnitude and direction. Notice that while FT enjoys comfortable variance between updates, LoRA is quite constrained. This low variance is indicative of the aforementioned entanglement of direction and magnitude– because LoRA has so fewer a number of parameters to work with, it loses expressiveness in what it can do in its updates.

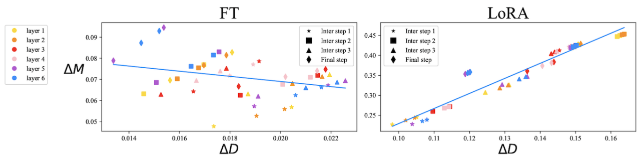

Figure 1. Full FT vs. LoRA Discrepancy (Credit to *DoRA, Liu et. al 2024*)

DoRA: Weight-Decomposed Low-Rank Adaptation, introduced in 2024, addresses this limitation directly. DoRA modifies LoRA by decoupling the update into two components: *direction* and *magnitude*. Instead of treating the low-rank matrix product AB as a complete update to the base weights W_0, DoRA constructs an intermediate matrix:

$$V' = V + AB, \tag{1}$$

where V is a direction vector initialized from the base model weights (e.g., W_0), and AB is the low-rank LoRA update applied only to the direction. The final weight matrix is then given by:

$$W = m \cdot \frac{V'}{\|V'\|_c}, \tag{2}$$

where:

- $\frac{V'}{\|V'\|_c}$ represents the **normalized direction**, and
- $m \in \mathbb{R}$ is a **learnable magnitude** scalar trained via standard backpropagation.

This reparameterization allows DoRA to update the direction (via LoRA) and the magnitude (via direct gradient descent) independently. It improves the expressiveness of the update and better approximates full fine-tuning behavior while remaining parameter-efficient.

Conceptually, this is similar to how *LayerNorm* operates: LayerNorm normalizes the direction of an activation vector and learns a separate scaling factor, thereby disentangling the statistical shape from its amplitude. Likewise, DoRA normalizes the update direction and learns its scale independently, improving both stability and capacity for generalization.

This seemingly simple change—separating what direction to move in from how far to move—proves surprisingly effective. In the sections that follow, we describe our implementation of this decomposition and present empirical evidence supporting its benefits.

2 Targeted Evaluation: DoRA vs. LoRA

The researchers in the original DoRA paper noted that their algorithm proved a much better approximation of the high variance that full FT was able to achieve with respect to magnitude and direction. As a result, DoRA achieved significantly higher accuracy on a variety of downstream tasks, compared to LoRA.

Our primary goal was to replicate these results and empirically verify that the alternative PEFT algorithm proposed by DoRA led to quantifiable improvements in commonsense reasoning tasks. We sought to verify the hypothesis that decoupling magnitude and direction allowed for better model flexibility in updating each of them.

3 Methodology

To validate DoRA's performance, we trained a LLaMA-7B model using both LoRA and DoRA on commonsense reasoning datasets. All training was done on an A-100 GPU. In the original paper, the authors trained on an aggregated commonsense reasoning dataset which was a combination of 8 different sub tasks. Rather than aggregating all datasets in this way we trained and tested on three: BoolQ, WinoGrande, and HellaSwag. Additionally, we trained and validated the models separately on each dataset, resulting in six total training runs (3 for LoRA, 3 for DoRA). We did this to avoid running out of memory during training. Each dataset involves a distinct task: BoolQ contains true/false classification, HellaSwag contains sentence completion, and WinoGrande contains fill-in-the-blank.

For re-implementing LoRA, we used publicly available implementations. For DoRA, we adapted LoRA's structure to incorporate the weight decomposition described in the introduction.

4 Results & Analysis

PEFT Strategy	BoolQ	HellaSwag	WinoGrande
Paper's LoRA	67.5	83.4	80.4
Our LoRA	63.5	41.2	74.7
Paper's DoRA	69.7	87.2	81.9
Our DoRA	65.8	51.2	79.5

Figure 2. Llama-7B Evaluation results on BoolQ, HellaSwag, and WinoGrande

From Figure 2, we can see that our approach yielded accuracies similar to those in the original paper on BoolQ and Winogrande, but on HellaSwag our accuracy was much lower. We suspect that this is due to HellaSwag containing a more complex task (correctly completing a sentence by choosing one of 4 offered endings), whereas BoolQ and Winogrande are binary tasks (true/false and fill-in-the-blank given only two options). We believe that all of our accuracies are lower than the paper's because we were unable to train on the full dataset used in the paper, as explained in methodology.

Regardless, the most important trend we see in this data is that it supports the paper's main finding: DoRA outperforms LoRA on all three datasets. This is demonstrated more clearly in Figure 3, below. This gives us confidence that DoRA is a viable alternative to LoRA when performing PEFT for commonsense reasoning tasks.

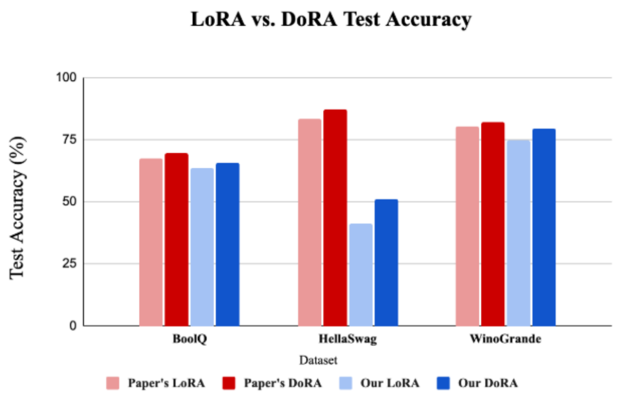

Figure 3. Llama-7B Evaluation results on BoolQ, HellaSwag, and WinoGrande

5 Reflections

As we re-implemented DoRA, immediately came to understand the breadth of computational resources required to train large models. We were surprised to learn that not even 40GB of GPU RAM was enough to post-train LLaMA-7B on congolomerated datasets used in the original paper and had to make adjustments: we trained on each dataset individually, one-by-one, instead of putting them together as the DoRA researchers had. Even so, each training-evaluation cycle nearly maxed out the available GPU memory. This puts into perspective how resource-intensive fine-tuning can be, even after making parameter-efficient approximations. Nevertheless, we found our results to be an encouraging sign that PEFT is possible in practical settings.

We identify several potential improvements that could be explored to further DoRA's objective, including adaptive rank parameterization techniques and hybrid modeling. We think that assigning the same low r for *all* weight matrices to be updated was perhaps too simplistic– more could be done to determine which matrices might have a higher ground truth rank. We suggest an analysis of the eigenstructure or singular values of the weight matrices as a means of informing our rank selection.

Acknowledgements

This project received support as part of CS 4782: Introduction to Deep Learning, instructed by professors Kilian Q. Weinberger and Jennifer J. Sun, at Cornell University.

References

[1] Liu, S.-Y., Wang, C.-Y., Yin, H., Molchanov, P., Wang, Y.-C. F., Cheng, K.-T., & Chen, M.-H. (2024). DoRA: Weight-Decomposed Low-Rank Adaptation. *arXiv preprint arXiv:2402.09353.*

[2] Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., & Chen, W. (2021). LoRA: Low-Rank Adaptation of Large Language Models. *arXiv preprint arXiv:2106.09685.*

[3] DoRA GitHub Repository

[4] PEFT GitHub Repository

[5] LLM-Adapters GitHub Repository (datasets)