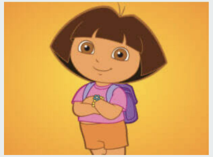
DoRA: Weight-Decomposed Low-Rank Adaptation



Motivation

- Parameter efficient finetuning is great!
  - It lets us finetune efficiently
  - And has no overheads during inference

- But there's a gap between FT and LoRA
  - This is attributed to fewer trainable parameters
  - But is that all there is to say about this?

- Maybe LoRA has certain patterns of updates that are different from FT…

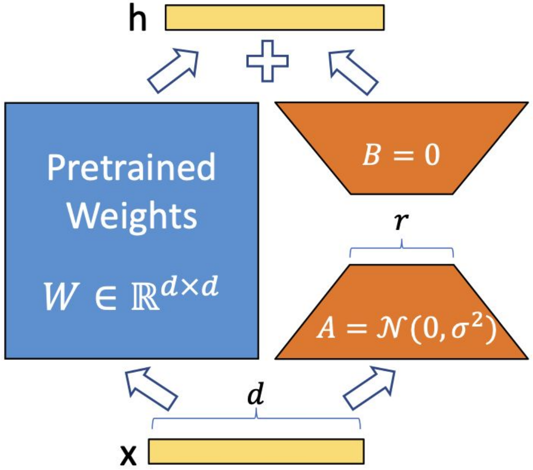

h
Pretrained
Weights
$W \in \mathbb{R}^{d \times d}$
$B = 0$
r
$A = \mathcal{N}(0, \sigma^2)$
d
x

Figure 1: Our reparametrization. We only train A and B.

Contributions

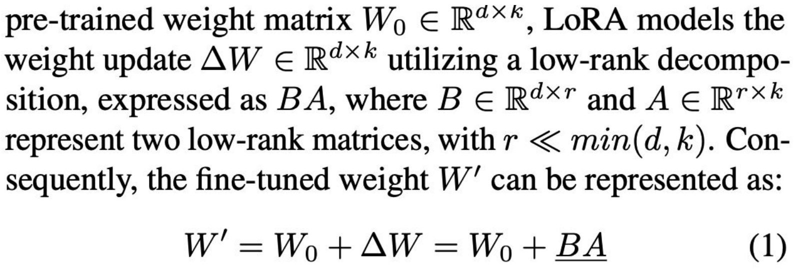
- Introduce DoRA, which achieves a performance closer to FT

- Weight decomposition analysis
  - Discover learning patterns for FT and LoRA that explains the difference in performance

- Empirical results supporting the above

LoRA : Recap

- Weight matrix has an intrinsic low rank



pre-trained weight matrix $W_0 \in \mathbb{R}^{d \times k}$, LoRA models the weight update $\Delta W \in \mathbb{R}^{d \times k}$ utilizing a low-rank decomposition, expressed as BA, where $B \in \mathbb{R}^{d \times r}$ and $A \in \mathbb{R}^{r \times k}$ represent two low-rank matrices, with $r \ll min(d, k)$. Consequently, the fine-tuned weight W' can be represented as:

$$W' = W_0 + \Delta W = W_0 + \underline{BA} \qquad (1)$$

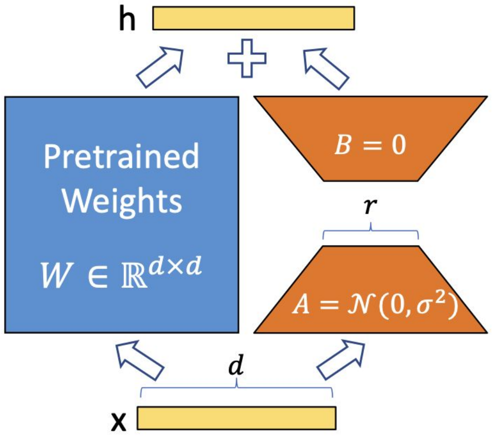

h
Pretrained Weights
$W \in \mathbb{R}^{d \times d}$
$B = 0$
r
$A = \mathcal{N}(0, \sigma^2)$
d
x

Figure 1: Our reparametrization. We only train A and B.

DoRA

- Let's decompose the weight matrix into its magnitude and direction
- Decomposition: $W = m\frac{V}{||V||_c} = ||W||_c\frac{W}{||W||_c}$
- Train magnitude separately
- Direction is trained using LoRA
  - More parameters

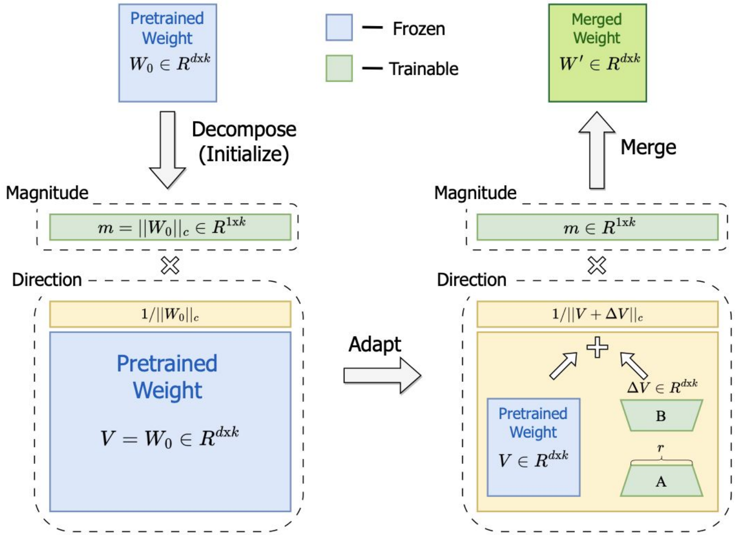


Figure 1. An overview of our proposed DoRA, which decomposes the pre-trained weight into *magnitude* and *direction* components for fine-tuning, especially with LoRA to efficiently update the direction component. Note that $|| \cdot ||_c$ denotes the vector-wise norm of a matrix across each column vector.

Inspiration: Weight Normalization

- (Salimans & Kingma, 2016)

- Reparametrization in this way
  - Magnitude and direction

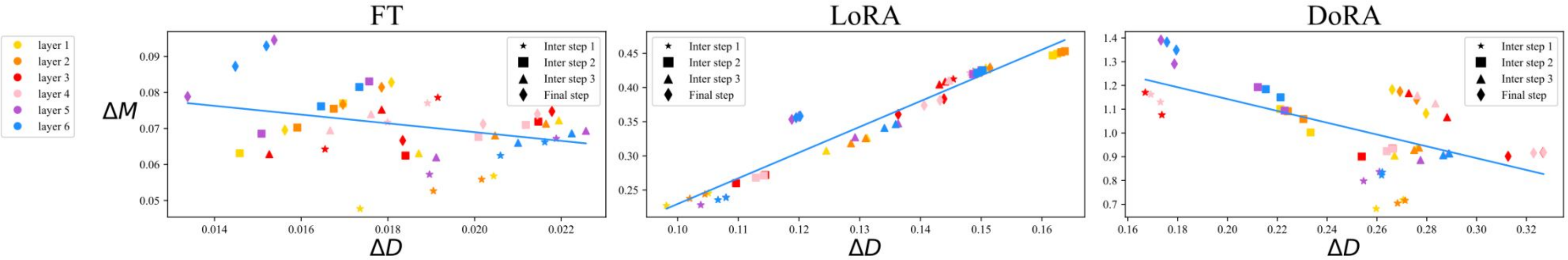
- Training from scratch

Weight decomposition Analysis

- Decomposition: $$W = m\frac{V}{||V||_c} = ||W||_c\frac{W}{||W||_c}$$

- Magnitude and directional difference:
$$\Delta M^t_{\mathrm{FT}} = \frac{\sum_{n=1}^{k} |m^{n,t}_{\mathrm{FT}} - m^n_0|}{k}$$
$$\Delta D^t_{\mathrm{FT}} = \frac{\sum_{n=1}^{k} (1 - \mathbf{cos}(V^{n,t}_{\mathrm{FT}}, W^n_0))}{k}$$

- VL-BART model finetuned on 4 image-text tasks
- Only query/value weight matrix in self-attention
- Different checkpoints



(a) (b) (c)

Figure 2. Magnitude and direction updates of (a) FT, (b) LoRA, and (c) DoRA of the query matrices across different layers and intermediate steps. Different markers represent matrices of different training steps and different colors represent the matrices of each layer.

Discussion

- FT slightly negative slope
- Hypothesis: pretrained weights already know a lot. It’s enough to just change one thing

- LoRA has a consistently positive slope
- LoRA cannot learn more nuanced adjustments

- DoRA also has a negative slope…
- …yay?

Further motivation DoRA

- Allows directional adaptation with LoRA and magnitude learning separately
- Instead of together as in LoRA

- Can be merged with pretrained weights before inference - no overhead latency

Gradient Analysis

- Can be shown that if directional update is lower, then magnitude update is higher (given difference norm is the same)
- This is good for stability and optimization
- Summary of proof:
  - If directional update is lower, cosine similarity of gradient and weight matrix is lower
  - But if gradient norm is the same, then this means that magnitude update is higher

$$W' = \underline{m}\frac{V + \Delta V}{||V + \Delta V||_c} = \underline{m}\frac{W_0 + \underline{BA}}{||W_0 + \underline{BA}||_c}$$

$$\nabla_{V'}\mathcal{L} = \frac{m}{||V'||_c}\left(I - \frac{V'V'^{\mathbf{T}}}{||V'||_c^2}\right)\nabla_{W'}\mathcal{L}$$

$$\nabla_m\mathcal{L} = \frac{\nabla_{W'}\mathcal{L} \cdot V'}{||V'||_c}$$

Training overhead

- We don’t like to backpropagate to more things

$$W' = \underline{m}\frac{V + \Delta V}{||V + \Delta V||_c} = \underline{m}\frac{W_0 + \underline{BA}}{||W_0 + \underline{BA}||_c}$$

- This denominator is extra → let’s ignore it

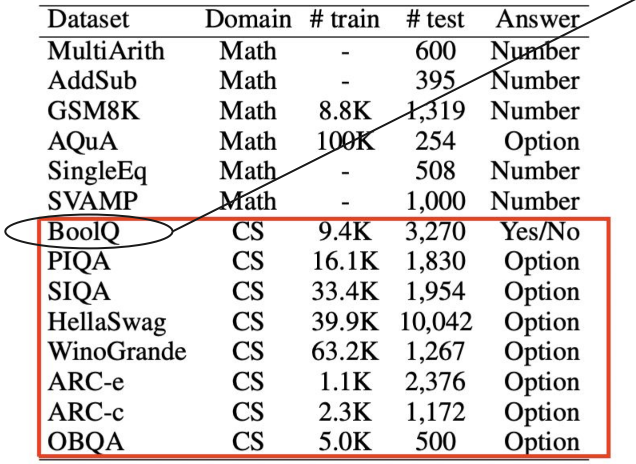
- It will still have actual norm, but won’t receive gradients

$$\nabla_{V'}\mathcal{L} = \frac{m}{||V'||_c}\left(I - \frac{V'V'^{\mathbf{T}}}{||V'||_c^2}\right)\nabla_{W'}\mathcal{L} \quad \longrightarrow \quad \nabla_{V'}\mathcal{L} = \frac{m}{C}\nabla_{W'}\mathcal{L} \text{ where } C = ||V'||_c$$

$$\nabla_m\mathcal{L} = \frac{\nabla_{W'}\mathcal{L} \cdot V'}{||V'||_c}$$

Experiments - Commonsense Reasoning

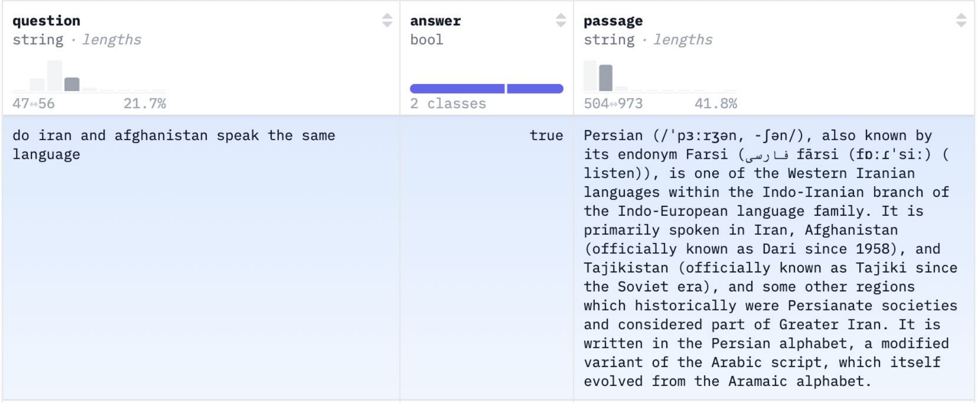
Datasets



Dataset	Domain	# train	# test	Answer
MultiArith	Math	-	600	Number
AddSub	Math	-	395	Number
GSM8K	Math	8.8K	1,319	Number
AQuA	Math	100K	254	Option
SingleEq	Math	-	508	Number
SVAMP	Math	-	1,000	Number
BoolQ	CS	9.4K	3,270	Yes/No
PIQA	CS	16.1K	1,830	Option
SIQA	CS	33.4K	1,954	Option
HellaSwag	CS	39.9K	10,042	Option
WinoGrande	CS	63.2K	1,267	Option
ARC-e	CS	1.1K	2,376	Option
ARC-c	CS	2.3K	1,172	Option
OBQA	CS	5.0K	500	Option

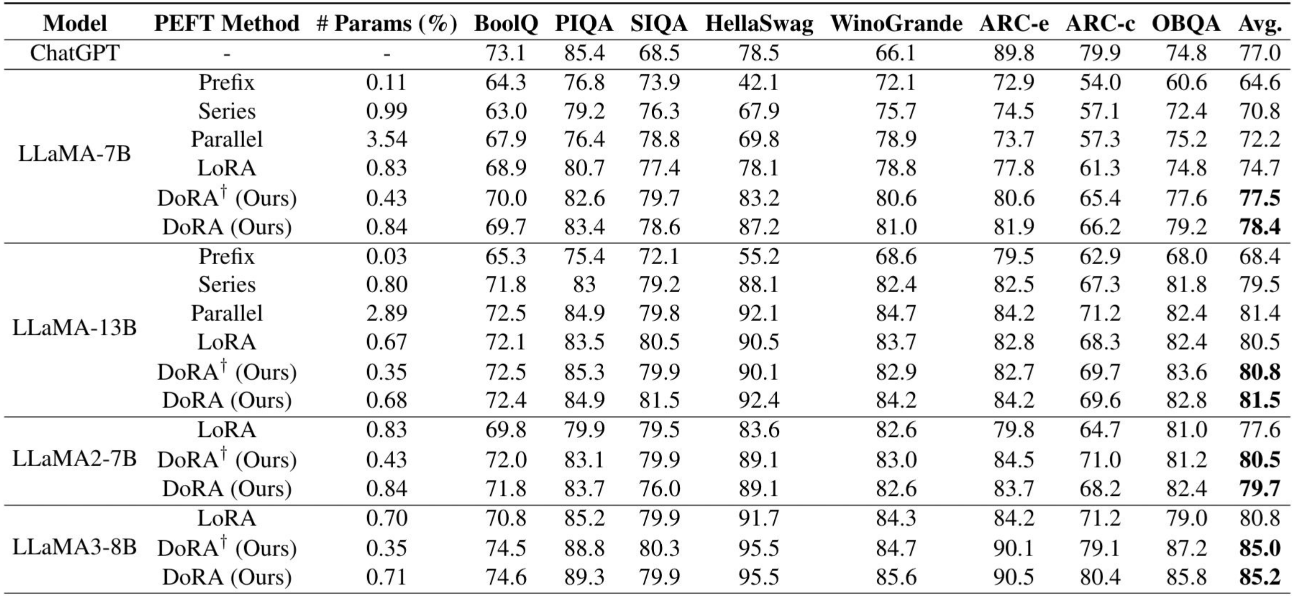
Table 2: Details of datasets being evaluated. Math: arithmetic reasoning. CS: commonsense reasoning.

BoolQ



question string · *lengths*	answer bool	passage string · *lengths*
do iran and afghanistan speak the same language	true	Persian (/ˈpɜːrʒən, -ʃən/), also known by its endonym Farsi (فارسی fārsi (fɒːɾˈsiː) (listen)), is one of the Western Iranian languages within the Indo-Iranian branch of the Indo-European language family. It is primarily spoken in Iran, Afghanistan (officially known as Dari since 1958), and Tajikistan (officially known as Tajiki since the Soviet era), and some other regions which historically were Persianate societies and considered part of Greater Iran. It is written in the Persian alphabet, a modified variant of the Arabic script, which itself evolved from the Aramaic alphabet.

Experiments - Commonsense Reasoning



Model	PEFT Method	# Params (%)	BoolQ	PIQA	SIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA	Avg.
ChatGPT	-	-	73.1	85.4	68.5	78.5	66.1	89.8	79.9	74.8	77.0
LLaMA-7B	Prefix	0.11	64.3	76.8	73.9	42.1	72.1	72.9	54.0	60.6	64.6
	Series	0.99	63.0	79.2	76.3	67.9	75.7	74.5	57.1	72.4	70.8
	Parallel	3.54	67.9	76.4	78.8	69.8	78.9	73.7	57.3	75.2	72.2
	LoRA	0.83	68.9	80.7	77.4	78.1	78.8	77.8	61.3	74.8	74.7
	DoRA† (Ours)	0.43	70.0	82.6	79.7	83.2	80.6	80.6	65.4	77.6	**77.5**
	DoRA (Ours)	0.84	69.7	83.4	78.6	87.2	81.0	81.9	66.2	79.2	**78.4**
LLaMA-13B	Prefix	0.03	65.3	75.4	72.1	55.2	68.6	79.5	62.9	68.0	68.4
	Series	0.80	71.8	83	79.2	88.1	82.4	82.5	67.3	81.8	79.5
	Parallel	2.89	72.5	84.9	79.8	92.1	84.7	84.2	71.2	82.4	81.4
	LoRA	0.67	72.1	83.5	80.5	90.5	83.7	82.8	68.3	82.4	80.5
	DoRA† (Ours)	0.35	72.5	85.3	79.9	90.1	82.9	82.7	69.7	83.6	**80.8**
	DoRA (Ours)	0.68	72.4	84.9	81.5	92.4	84.2	84.2	69.6	82.8	**81.5**
LLaMA2-7B	LoRA	0.83	69.8	79.9	79.5	83.6	82.6	79.8	64.7	81.0	77.6
	DoRA† (Ours)	0.43	72.0	83.1	79.9	89.1	83.0	84.5	71.0	81.2	**80.5**
	DoRA (Ours)	0.84	71.8	83.7	76.0	89.1	82.6	83.7	68.2	82.4	**79.7**
LLaMA3-8B	LoRA	0.70	70.8	85.2	79.9	91.7	84.3	84.2	71.2	79.0	80.8
	DoRA† (Ours)	0.35	74.5	88.8	80.3	95.5	84.7	90.1	79.1	87.2	**85.0**
	DoRA (Ours)	0.71	74.6	89.3	79.9	95.5	85.6	90.5	80.4	85.8	**85.2**

DoRA outperforms all baselines across LLaMA variants (7B/13B/2-7B/3-8B), with standout gains on LLaMA-7B (+3.7% vs. LoRA, surpassing ChatGPT)

On LLaMA-13B, DoRA matches Parallel adapter's accuracy while using 75% fewer parameters and no extra inference overhead

DoRA† (using half parameters) still beats LoRA by 1-4.2% across all models

Experiments - Commonsense Reasoning

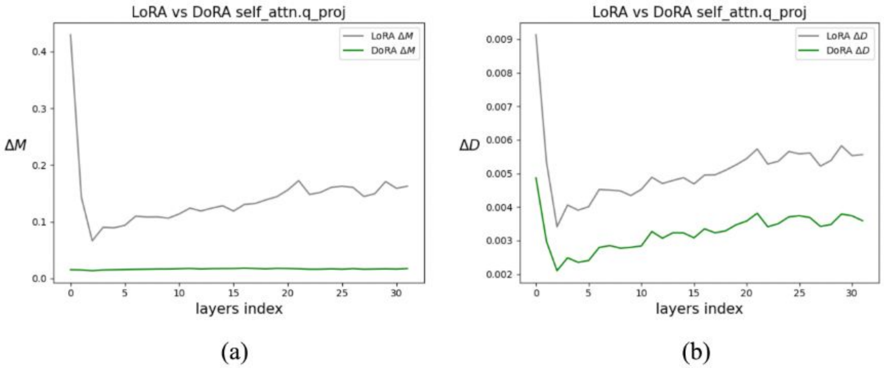


Figure 3. Magnitude (a) and direction (b) difference of LoRA/DoRA and the pre-trained weight of the query matrices across different layers.

Pre-trained weights were found to contain sufficient task knowledge - the authors hypothesized that subtle updates are optimal

Their analysis on LLaMA2-7B revealed an interesting pattern: DoRA maintains closer proximity to pre-trained weights vs. LoRA

This explains DoRA's effectiveness: its fine-grained update mechanism enables precise adjustments while preserving model knowledge

Experiments - Image/Video-Text Understanding

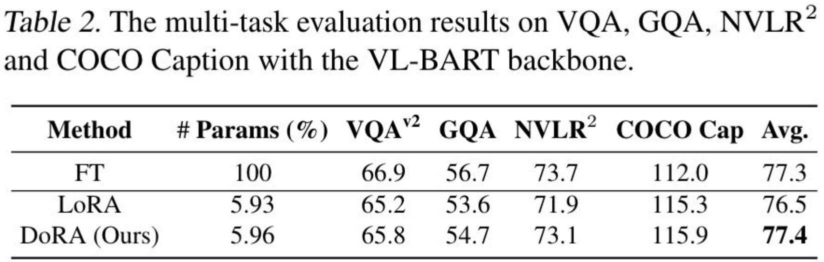


Table 2. The multi-task evaluation results on VQA, GQA, NVLR^2 and COCO Caption with the VL-BART backbone.

Method	# Params (%)	VQA^{v2}	GQA	NVLR^2	COCO Cap	Avg.
FT	100	66.9	56.7	73.7	112.0	77.3
LoRA	5.93	65.2	53.6	71.9	115.3	76.5
DoRA (Ours)	5.96	65.8	54.7	73.1	115.9	**77.4**

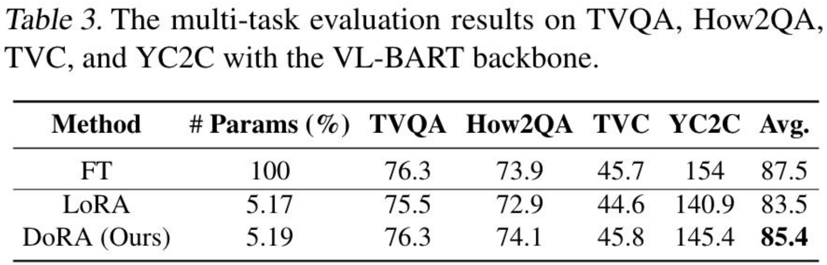


Table 3. The multi-task evaluation results on TVQA, How2QA, TVC, and YC2C with the VL-BART backbone.

Method	# Params (%)	TVQA	How2QA	TVC	YC2C	Avg.
FT	100	76.3	73.9	45.7	154	87.5
LoRA	5.17	75.5	72.9	44.6	140.9	83.5
DoRA (Ours)	5.19	76.3	74.1	45.8	145.4	**85.4**

DoRA not only outperformed LoRA (+1% on image tasks, +2% on video tasks), but also matched full fine-tuning accuracy with just a fraction of trainable parameters

Experiments - Visual Instruction Tuning

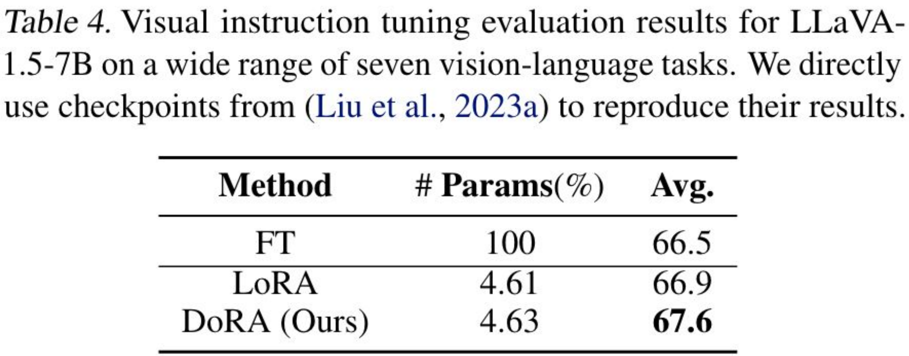


Table 4. Visual instruction tuning evaluation results for LLaVA-1.5-7B on a wide range of seven vision-language tasks. We directly use checkpoints from (Liu et al., 2023a) to reproduce their results.

Method	**# Params(%)**	**Avg.**
FT	100	66.5
LoRA	4.61	66.9
DoRA (Ours)	4.63	**67.6**

Despite LoRA already outperforming full fine-tuning, DoRA still showed consistent gains (+0.7% over LoRA, +1.1% over FT)

Experiments - Compatibility of DoRA with other LoRA variants

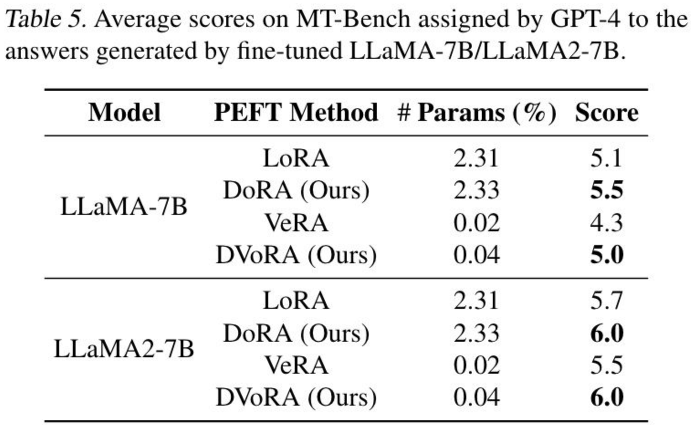


Table 5. Average scores on MT-Bench assigned by GPT-4 to the answers generated by fine-tuned LLaMA-7B/LLaMA2-7B.

Model	PEFT Method	# Params (%)	Score
LLaMA-7B	LoRA	2.31	5.1
	DoRA (Ours)	2.33	**5.5**
	VeRA	0.02	4.3
	DVoRA (Ours)	0.04	**5.0**
LLaMA2-7B	LoRA	2.31	5.7
	DoRA (Ours)	2.33	**6.0**
	VeRA	0.02	5.5
	DVoRA (Ours)	0.04	**6.0**

Test DoRA’s compatibility with VeRA by combining DoRA's directional updates with VeRA's low-parameter, shared-matrix approach for efficiency.

Outperforms LoRA and VeRA on LLaMA-7B and LLaMA2-7B (10K Alpaca dataset).

Experiments - Robustness of DoRA towards different rank settings

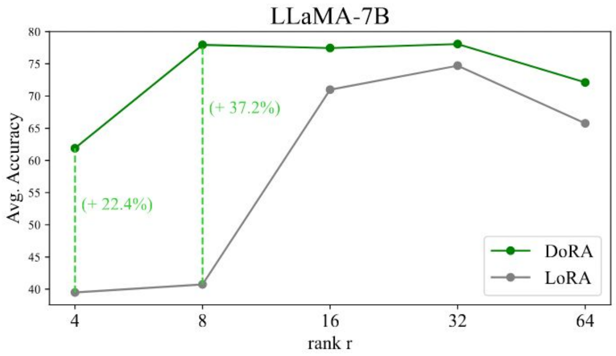


Figure 5. Average accuracy of LoRA and DoRA for varying ranks for LLaMA-7B on the commonsense reasoning tasks.

DoRA consistently outperforms LoRA at all ranks. Significant gap at lower ranks:

- For r = 4, DoRA achieves 61.89% accuracy vs. LoRA's 39.49%.
- For r = 8, DoRA reaches 77.96% accuracy vs. LoRA's 40.74%.

Experiments - Tuning Granularity Analysis

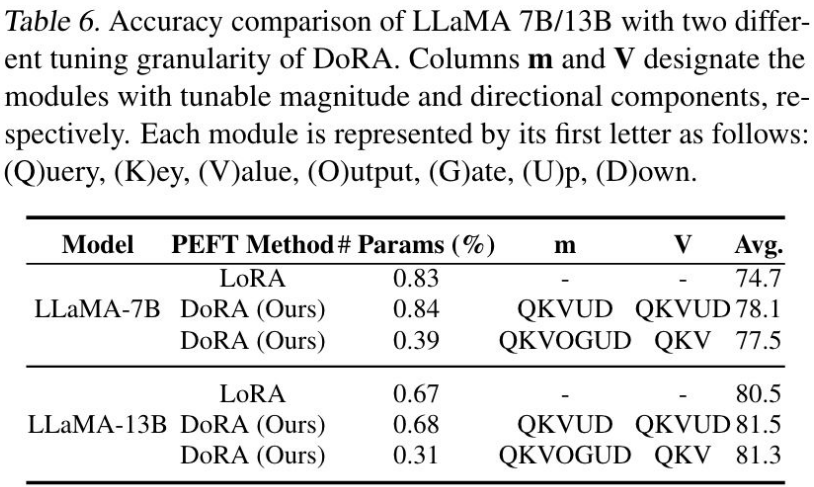


Table 6. Accuracy comparison of LLaMA 7B/13B with two different tuning granularity of DoRA. Columns **m** and **V** designate the modules with tunable magnitude and directional components, respectively. Each module is represented by its first letter as follows: (Q)uery, (K)ey, (V)alue, (O)utput, (G)ate, (U)p, (D)own.

Model	PEFT Method	# Params (%)	m	V	Avg.
	LoRA	0.83	-	-	74.7
LLaMA-7B	DoRA (Ours)	0.84	QKVUD	QKVUD	78.1
	DoRA (Ours)	0.39	QKVOGUD	QKV	77.5
	LoRA	0.67	-	-	80.5
LLaMA-13B	DoRA (Ours)	0.68	QKVUD	QKVUD	81.5
	DoRA (Ours)	0.31	QKVOGUD	QKV	81.3

DoRA can already achieve superior accuracy by updating only the directional and magnitude components of the multi-head layers and the magnitude of the MLP layers.

Method:

- Selective updates: QKV (direction & magnitude), MLP (magnitude only)

Key Results (Table 6):

- LLaMA-7B: +2.8% accuracy over LoRA
- LLaMA-13B: +0.8% accuracy over LoRA
- Efficiency: Uses <50% of LoRA’s parameters

Broader Impacts

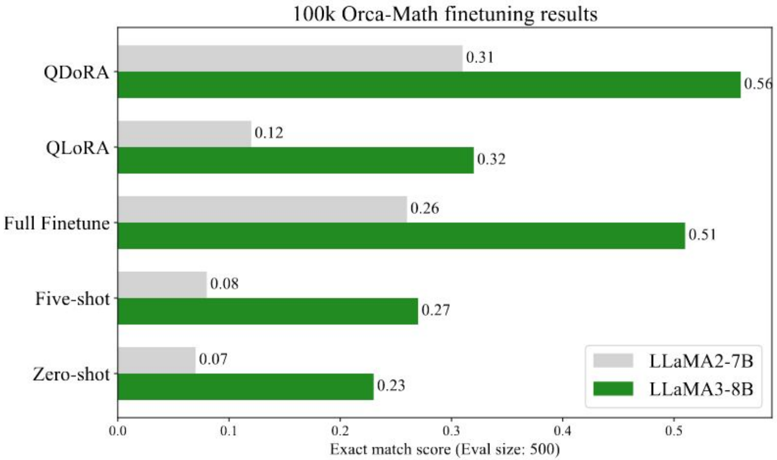


Figure 6. Accuracy comparison of LLaMA2-7B/LLaMA3-8B with QDoRA, QLoRA and FT on Orca-Math (Mitra et al., 2024).

QDoRA: Combines DoRA with QLoRA to improve memory efficiency in LLM fine-tuning, surpassing QLoRA by up to 0.23% and outperforming full fine-tuning with less GPU memory.

Text-to-Image: DoRA achieves more accurate and personalized results than LoRA, capturing unique features in generated images (e.g., frames, logos).

Thanks