RepLoRA: Reparameterizing Low-Rank Adaptation via the Perspective of Mixture of Experts

Tuan Truong*, Chau Nguyen*, Huy Nguyen*, Minh Le, Trung Le, Nhat Ho

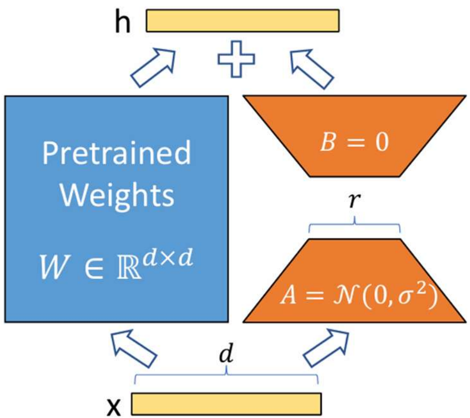
How do we utilize (huge) pre-trained models?

- Foundational models are increasingly demonstrating remarkable capabilities over a wide array of tasks.

- **Goal**: Effectively utilizing these pre-trained models for downstream tasks. However, adapting these models via **full fine-tuning** presents significant limitations: **high computational cost**, **overfitting**, **storage overhead**…

Low-rank Adaptation (LoRA)

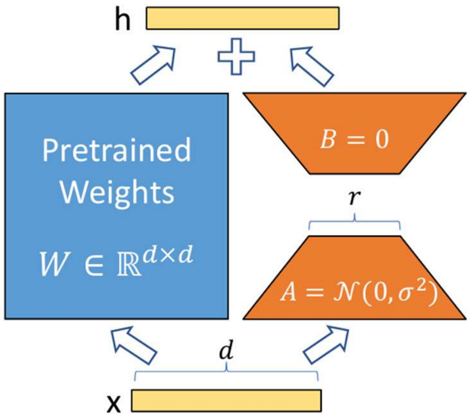
- A parameter-efficient fine-tuning technique.
- Only learn two low-rank matrices $\boldsymbol{A}$, and $\boldsymbol{B}$ instead of full weight matrix $\boldsymbol{W}_0$.



$$\hat{y} = \boldsymbol{W}'\boldsymbol{x} = \boldsymbol{W}_0\boldsymbol{x} + \boldsymbol{B}\boldsymbol{A}\boldsymbol{x}.$$

Low-rank Adaptation (LoRA)

- Typically, LoRA is applied in **attention** module, specifically the **query** and **value** weights.
- Despite its successes**, theoretical understanding of LoRA has remained limited**, hindering our ability to optimize its performance further.



$$\hat{y} = \boldsymbol{W}'\boldsymbol{x} = \boldsymbol{W}_0\boldsymbol{x} + \boldsymbol{B}\boldsymbol{A}\boldsymbol{x}.$$

Mixture of Experts (MoE)

- **Mixture of Experts model:** An MoE model consists of a group of N' expert networks $f_i,\ \ i\ \in [N']$, and a gate function G. The output is expressed as:

$$\hat{\mathbf{y}} = \sum_{i=1}^{N'} G(\boldsymbol{x})_i \cdot f_i(\boldsymbol{x}) = \sum_{i=1}^{N'} \frac{\exp\left(s_i(\boldsymbol{x})\right)}{\sum_{j=1}^{N'} \exp\left(s_j(\boldsymbol{x})\right)} \cdot f_i(\boldsymbol{x}),$$

where $G(\boldsymbol{x}) = softmax(s_1(\boldsymbol{x}), \dots, s_{N'}(\boldsymbol{x}))$

LoRA and MoE

- **Each attention head** is equivalent to **multiple MoE models**.
- Let $\boldsymbol{X} = [\boldsymbol{x}_1^\top, \ldots, \boldsymbol{x}_N^\top]^\top$ denote the concatenated input embedding. The experts and score functions are defined as follows with $i \in [N], j \in [N]$:

$$f_j(\mathbb{X}) = {\boldsymbol{W}_l^V}^\top \boldsymbol{E}_j \mathbb{X} = {\boldsymbol{W}_l^V}^\top \boldsymbol{x}_j,$$

$$s_{i,j}(\mathbb{X}) = \frac{\mathbb{X}^\top \boldsymbol{E}_i^\top \boldsymbol{W}_l^Q {\boldsymbol{W}_l^K}^\top \boldsymbol{E}_j \mathbb{X}}{\sqrt{d_v}} = \frac{\boldsymbol{x}_i^\top \boldsymbol{W}_l^Q {\boldsymbol{W}_l^K}^\top \boldsymbol{x}_j}{\sqrt{d_v}},$$

LoRA and MoE

- When LoRA is applied to query and value matrices, it refines these experts and score functions with updates:

$$\tilde{f}_j(\mathbb{X}) = (\boldsymbol{W}_l^V + \boldsymbol{B}_{V,l}\boldsymbol{A}_{V,l})^\top \boldsymbol{E}_j \mathbb{X},$$

$$\tilde{s}_{i,j}(\mathbb{X}) = \frac{\mathbb{X}^\top \boldsymbol{E}_i^\top (\boldsymbol{W}_l^Q + \boldsymbol{B}_{Q,l}\boldsymbol{A}_{Q,l}) \boldsymbol{W}_l^{K\top} \boldsymbol{E}_j \mathbb{X}}{\sqrt{d_v}},$$

- LoRA effectively fine-tunes the pre-trained MoE models contained within each attention head by incorporating low-rank modifications to both the expert and the score functions.

RepLoRA: Reparameterizing Low-Rank Adaptation

- We show that simple **reparameterization** of the LoRA matrices can notably accelerate the low-rank matrix estimation process.
- LoRA in attention:

$$\boldsymbol{W}'_Q = \boldsymbol{W}_Q + \boldsymbol{B}_Q \boldsymbol{A}_Q \qquad \boldsymbol{W}'_V = \boldsymbol{W}_V + \boldsymbol{B}_V \boldsymbol{A}_V,$$

- RepLoRA innovatively reparameterizes $\boldsymbol{A}$ and $\boldsymbol{B}$, modeling them as outputs of two **shared MLPs**:

$$[\boldsymbol{A}_Q, \boldsymbol{A}_V] = g_{\theta_{\boldsymbol{A}}}(\boldsymbol{A}) \qquad [\boldsymbol{B}_Q, \boldsymbol{B}_V] = g_{\theta_{\boldsymbol{B}}}(\boldsymbol{B}),$$

- We implement $\boldsymbol{A}$ and $\boldsymbol{B}$ as **diagonal matrices** to ensure simplicity parameter efficiency.

Theoretical Justifications of RepLoRA

- We prove that estimating parameters in RepLoRA is statistically efficient in terms of the number of data points.

Model	Parameter estimation rate	Number of data
LoRA	$O(1/\log(n)^{\tau})$	**Exponential** $\exp(\varepsilon^{-\tau})$
RepLoRA	$O(\sqrt[4]{\log(n)/n})$	**Polynomial** ε^{-4}

Experiments

Commonsense reasoning

Table 1. Top-1 Accuracy and PPT on commonsense datasets. The accuracies are reported with `LLaMA-7B` and `LLaMA-13B`.

Model	**Method**	**#Params (%)**	BoolQ	PIQA	SIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA	**AVG**	**PPT**
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	0.83
	LoRA	0.83	67.2	79.4	76.6	78.3	78.4	77.1	61.5	74.2	74.1	1.70
	Adapter	0.99	63.0	79.2	76.3	67.9	75.7	74.5	57.1	72.4	70.8	1.74
	DoRA	0.98	69.7	83.4	78.6	**87.2**	81.0	81.9	66.2	79.2	78.4	1.81
	RepLoRA	1.01	**71.8**	**84.1**	**79.3**	85.2	**83.3**	**82.4**	**66.2**	**81.2**	**79.1**	**1.96**
LLaMA-13B	Prefix	0.03	65.3	75.4	72.1	55.2	68.6	79.5	62.9	68.0	68.4	0.79
	LoRA	0.67	71.7	82.4	79.6	90.4	83.6	83.1	68.5	82.1	80.2	2.15
	Adapter	0.80	71.8	83.0	79.2	88.1	82.4	82.5	67.3	81.8	79.5	1.80
	DoRA	0.68	72.4	84.9	81.5	**92.4**	84.2	84.2	69.6	82.8	81.5	2.19
	RepLoRA	0.99	**73.1**	**85.2**	**84.7**	91.1	**85.9**	**84.7**	**73.4**	**85.6**	**82.9**	**2.60**

Experiments

Image classification

Table 2: Classification performance on FGVC datasets.

Method	CUB-200 -2011	NABirds	Oxford Flowers	Stanford Dogs	Stanford Cars	AVG	PPT
FFT	87.3	82.7	98.8	89.4	84.5	88.5	-
LoRA	84.6	78.2	98.9	85.1	77.1	84.8	0.82
Adapter	87.1	84.3	98.5	89.8	68.6	85.6	0.84
Prefix	87.5	82.0	98.0	74.2	**90.2**	86.3	0.85
RepLoRA	**89.1**	**86.1**	**99.3**	**91.2**	87.6	**90.7**	**0.90**

Table 3. Performance on VTAB-1K with ViT-B/16 pre-trained on ImageNet-21K.

	Natural							Specialized				Structured									
Method	CIFAR100	Caltech101	DTD	Flower102	Pets	SVHN	Sun397	Camelyon	EuroSAT	Resisc45	Retinopathy	Clevr-Count	Clevr-Dist	DMLab	KITTI	dSpr-Loc	dSpr-Ori	sNORB-Azim	sNORB-Ele	**AVG**	**PPT**
FFT	68.9	87.7	64.3	97.2	86.9	87.4	38.8	79.7	95.7	84.2	73.9	56.3	58.6	41.7	65.5	57.5	46.7	25.7	29.1	65.5	-
LoRA	67.1	91.4	69.4	98.2	90.4	85.3	54	84.9	95.3	84.4	73.6	82.9	69.2	49.8	78.5	75.7	47.1	31	44.0	72.2	0.72
Adapter	69.2	90.1	68	98.8	89.9	82.8	54.3	84	94.9	81.9	75.5	80.9	65.3	48.6	78.3	74.8	48.5	29.9	41.6	71.4	0.71
Prefix	**75.5**	90.7	65.4	96.6	86	78.5	46.7	79.5	95.1	80.6	74.0	69.9	58.2	40.9	69.5	72.4	46.8	23.9	34.4	67.6	0.73
RepLoRA	73.2	**94.1**	**73.3**	**99.3**	**94.4**	**89.1**	**58.9**	**89.2**	**97.5**	**87.9**	**77.8**	**85.1**	**72.6**	**55.7**	**81.2**	**81.7**	**49.2**	**35.7**	**47.3**	**75.9**	**0.74**

Experiments

Video action recognition

Table 4: Performance on Video Action Recognition task.

				SSv2		HMDB51	
Method	**Model**	**Pretraining**	**#Params (M)**	**Acc@1**	**PPT**	**Acc@1**	**PPT**
FFT	Video Swin-B	Kinetics400	87.64	50.99	-	68.07	-
LoRA	Video Swin-B	Kinetics400	0.75	38.34	0.37	62.12	0.61
Adapter	Video Swin-B	Kinetics400	1.56	39.09	0.36	67.52	0.63
Prefix	Video Swin-B	Kinetics400	6.37	39.46	0.31	56.13	0.45
RepLoRA	Video Swin-B	Kinetics400	1.45	**46.12**	**0.41**	**68.23**	**0.64**

Experiments

Image and Video-Text understanding

Table 5. Performance on image-text tasks with VL−BART.

Method	#Params (%)	VQA^{v2}	GQA	NVLR^2	COCO Cap	AVG	PPT
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	0.99
DoRA	5.96	65.8	54.7	73.1	115.9	77.4	1.00
RepLoRA	6.02	66.5	55.4	**74.2**	**116.2**	**78.1**	**1.02**

Table 6. Performance on video-text tasks with VL−BART.

Method	#Params (%)	TVQA	How2QA	TVC	YC2C	AVG	PPT
FT	100	76.3	73.9	45.7	**154.0**	87.5	-
LoRA	5.17	75.5	72.9	44.6	140.9	83.5	1.06
DoRA	5.19	76.3	74.1	45.8	145.4	85.4	1.08
RepLoRA	5.30	**77.8**	**75.1**	**46.6**	151.6	**87.8**	**1.12**

Experiments

Sample Efficiency

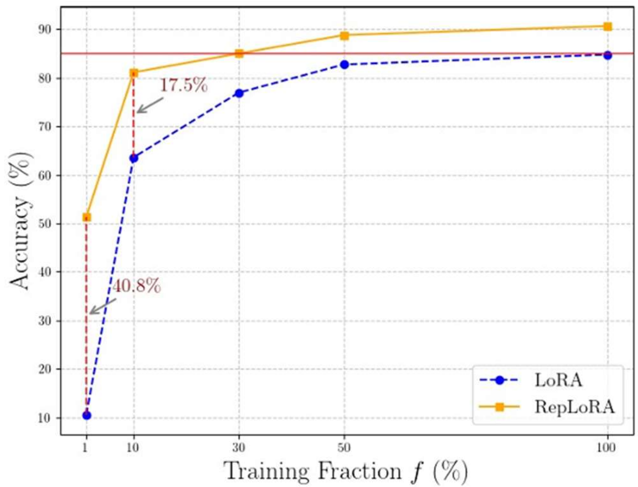


Figure 2: Sample Efficiency on FGVC Datasets. RepLoRA not only outperforms LoRA consistently but also achieves LoRA performance on a full dataset with only $f = 30\%$ training fraction.

Conclusion

- We introduce a novel theoretical framework that connects LoRA with MoE
- We build upon this framework and introduce RepLoRA, which:
  - ➢ Demonstrated its effectiveness on four diverse domains: image, video, text, and multi-modal tasks.
  - ➢ Is significantly more parameter-efficient than LoRA, both theoretically and practically

Thank you