Sparse Matrix in Large Language Model Fine-tuning

Presenter: ***Xuan Jiang***

***Collaborators*: *Haoze He*[1], *Billy Li*[1,2], *Heather Miller*[1]**

School of Computer Science, Carnegie Mellon University[1]

Two Sigma[2];

xuanj@mit.edu

Accepted by ICLR2025

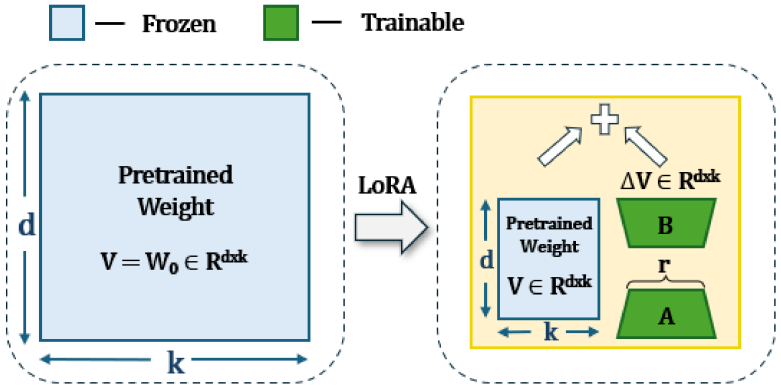
Research Questions?

- How to make LLM fine-tuning more memory efficient?
- How to speedup LLM fine-tuning?
- Which Matrix is more important in Attention? Q, K, or V?

Background

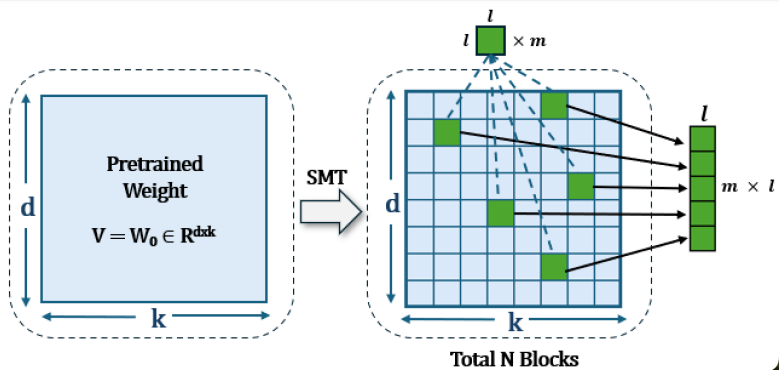
- As the **size** of LLMs increases, there is a pressing challenge to optimize the fine-tuning process for better **computational efficiency** and **memory utilization.**

Reparameterization category



Weakness: still indicate a performance gap between reparameterization methods and full fine-tuning. And it's facing scalability plateau

Specification category



Weakness : Current SOTA methods in specification category are **not efficient enough**. For instance: parameter selection can only be applied to layer levels; Require full backward propagation to compute full gradient;

SMT

- Identify submatrices

$$\mathrm{F}_{\hat{\theta}} = \frac{1}{N}\sum_{i=1}^{N} |\nabla_{\hat{\theta}} \log p_{\hat{\theta}}(y_i \,|x_i)|$$

$$\theta_i \in \{\, \theta_j \mid \hat{F}_{\theta_j} \geq sort(\hat{F}_{\theta_{[k]}}) \,\}$$

- where $\mathrm{F}_{\hat{\theta}}$ is the SMT calculation of gradient information, using the mean absolute gradient value for each sub-matrix parameter $\hat{\theta}$.
- N is The number of data samples; x_i is The i-th input data sample
- w_N and w_τ index for model weights coefficients. y_i is The ground-truth label corresponding to the i-th input;
- $\nabla_{\hat{\theta}} \log p_{\hat{\theta}}(y_i \,|x_i)$ The gradient of the log-likelihood of the model's output with respect to the selected sub-matrix parameter $\hat{\theta}$ for sample (x_i, y_i)

SMT Idea in One Sentence

- In SMT, we slice the pre-trained weight into N sub-matrices and only fine-tune selected M sub-matrices. The dimension of each sub-matrices is $l \times l$. The number of fine-tuning sub-matrices $m \ll N$.

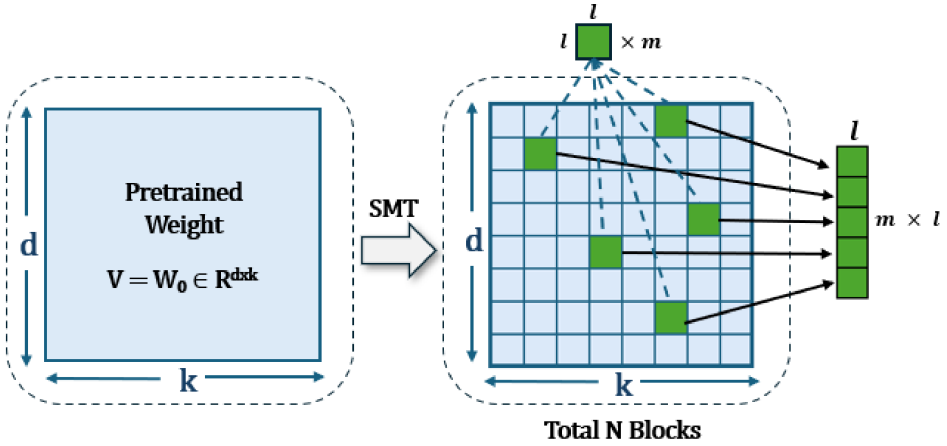


Figure 1. SMT high-level idea

Is It Efficient in Memory and Speedup?

- In SMT, SMT reduces the memory costs of the optimizer gradients to z% (z=0.5). Since SMT only updates selected sparse sub-matrices, partial gradients are stored to cut the memory costs of the Adam optimizer to z% (z=0.5).

  → *Offers 67% GPU memory savings, 76GB for 13B models*

- SMT also reduces the gradient step computation costs to z% since only partial gradient steps are required.

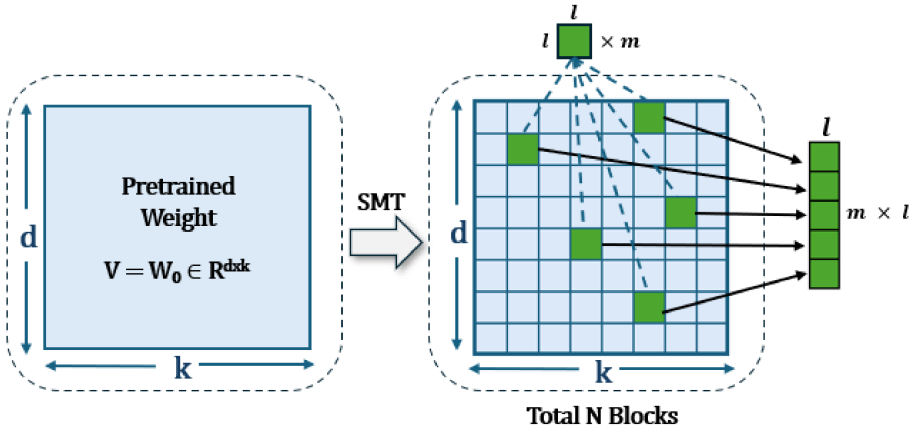


Figure 2. SMT high-level idea

- SMT significantly reduces backward computation costs to $z\%$ ($z = 0.5$) of those in Full Fine-tuning (FT) by calculating gradients for only a subset of the weights during backpropagation.

  → *Offers 14 times speedup comparing to full fine-tuning offload to CPU.*

- SMT reduces the activation memory costs for the in forward pass to $z\%$ ($z = 0.5$). Since SMT only computes the partial gradient, it saves the relevant portions of activations X necessary for the gradient calculation.

  → *Offers great memory savings, 50GB for 256 batch size.*

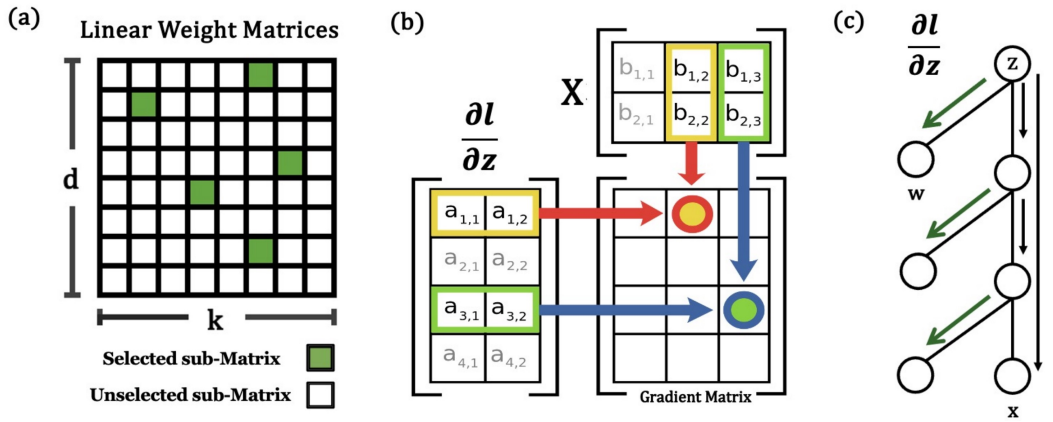


Figure 3. SMT Partial Backward Propagation

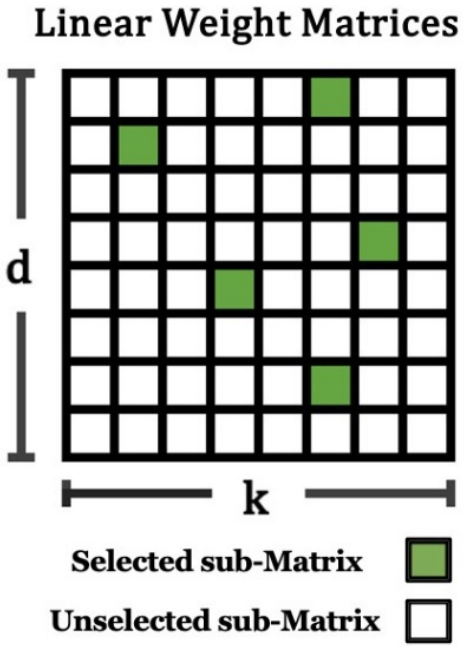
Is It Efficient in Memory and Speedup?

- Adaption methods need to maintain additional adapters, which require additional forward computation and memory cost.

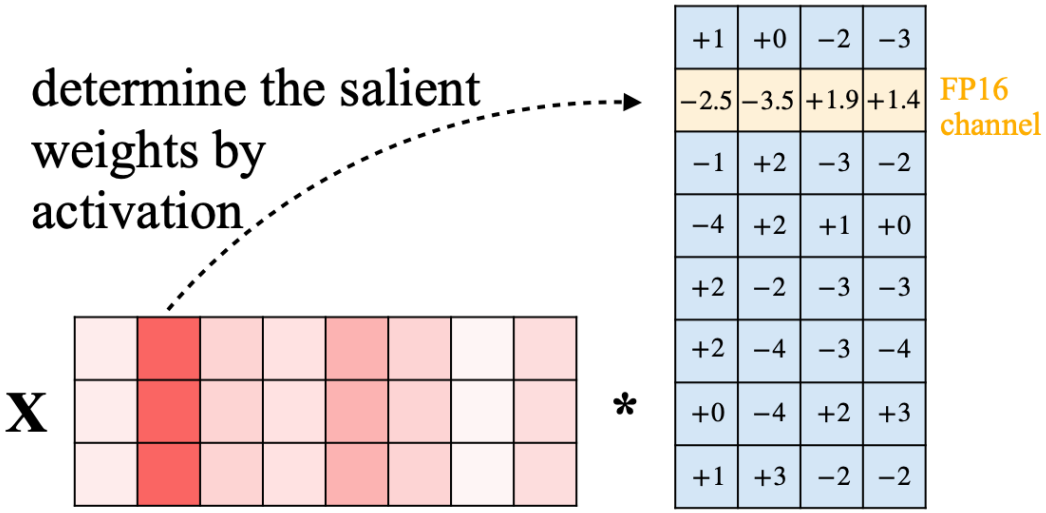
LLaMA-7B			
PEFT method	**#Params%**	**Time/s**	**Speedup**
Full Fine-tuning	100	243.84	1×
SMT	1.26	16.68	14.6×
LoRA	1.26	17.82	13.6×
DoRA	1.27	18.04	13.5×
SpIEL	1.26	25.45	9.6×

Speedup Compared to SFT and other SOTAs

How to Select Parameters?



Gradient-Aware Selection



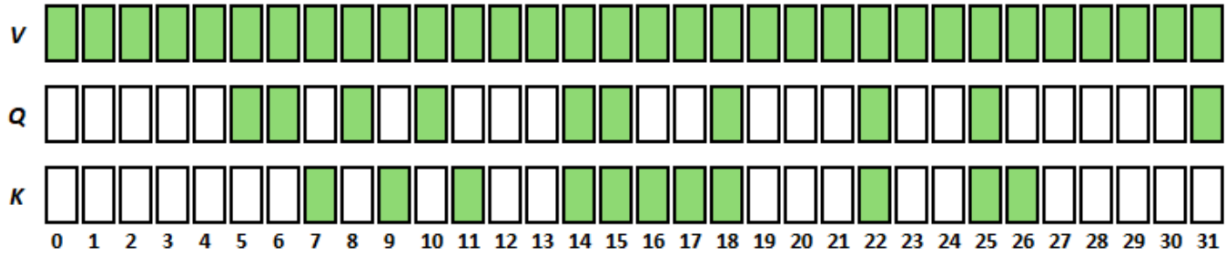
Activation-Aware Selection

- But how can we Know we select the correct Parameters? Any evidence other than test accuracy? (LLM Anatomy Cont.)

V versus Q, K

- QKV SMT assign all trainable parameters to QKV vectors and select sub-matrices automatically.
- 95.17% of the trainable parameters are automatically assigned to the V vectors by SMT. Fig.4 indicates that all V vectors have trainable parameters, while 22 out of 32 Q vectors and 21 out of 32 K vectors are completely frozen.

Figure 4: A visualization of trainable Q, K, V layers when fine-tuning 0.86% trainable parameters on LLaMA-7B. LLaMA-7B has 32 layers of MLPs, each contains a Q vector, a K vector, and a V vector. White layers are frozen and green layers contain trainable parameters.



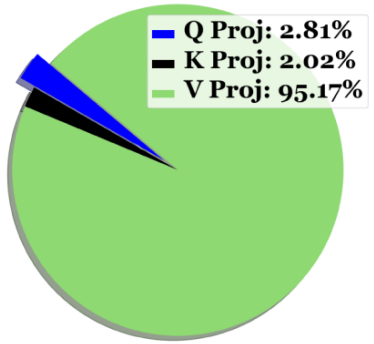


Figure 5: Distribution of trainable parameters among Q, K, V.

V versus Q, K

- We assign all 0.86% trainable parameters to only K, or only Q, or only V vectors (row1-3).
- The results show a significant performance gap when comparing the allocation of all trainable parameters to the V vectors versus the Q and K vectors.
- It hints that SMT can effectively select sub-matrices.

Table 6: K SMT, Q SMT, and V SMT assign all trainable parameters to only K, or only Q, or only V vectors respectively, and fine-tuned 0.86% of the parameters on LLaMA-7B using the Common-sense dataset. QKV SMT assign all trainable parameters to QKV vectors and select sub-matrices automatically.

Model	Param location	#Params%	BoolQ	PIQA	SIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA	AVG
LLaMA-7B	K SMT	0.84	65.5	79.1	76.2	88.3	73.2	80.3	60.8	68.0	73.9
	Q SMT	0.84	65.7	79.3	75.5	88.2	72.5	80.1	59.6	72.5	75.3
	V SMT	0.84	68.7	82.1	78.1	91.6	78.8	83.0	68.7	77.2	78.5
	QKV SMT	0.84	68.7	81.7	78.3	91.6	78.8	84.1	68.7	77.4	78.7

Why are V More Crucial?

- We observe that the gradients of the V vectors are significantly larger than those of the Q and K vectors, with the V vector gradients being up to 10 times greater in most layers. This larger gradient leads to more substantial updates, making fine-tuning the V vectors more effective.

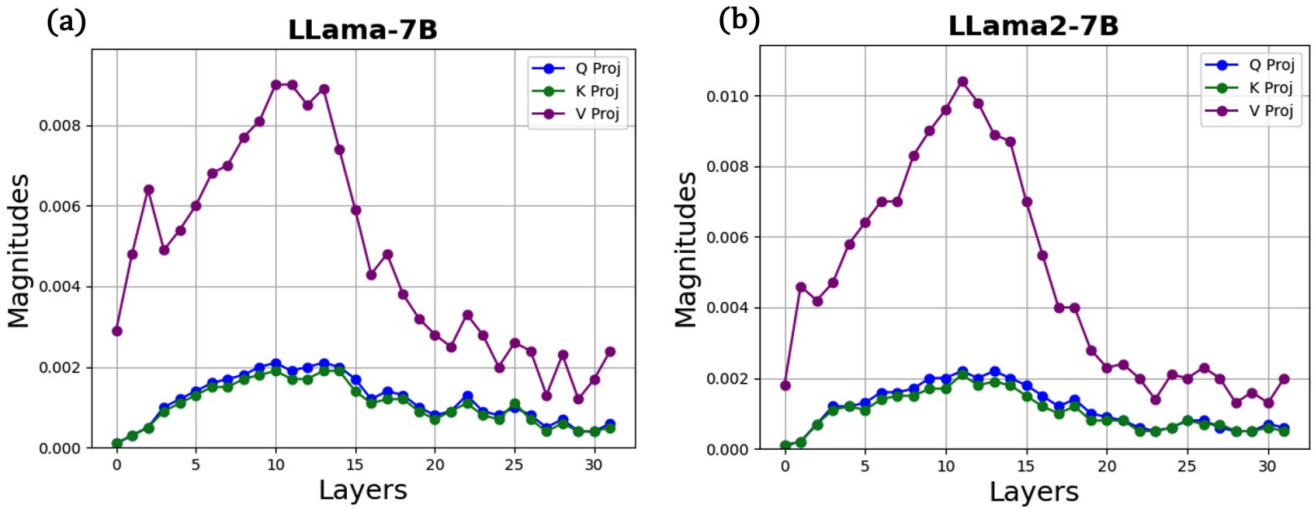


Figure 5. The magnitude of the gradient for the Q, K, and V vectors at each layer in LLMs.

Why V has Larger Gradient?

- We observe that the gradients of the V vectors are significantly larger than those of the Q and K vectors, with the V vector gradients being up to 10 times greater in most layers. This larger gradient leads to more substantial updates, making fine-tuning the V vectors more effective.

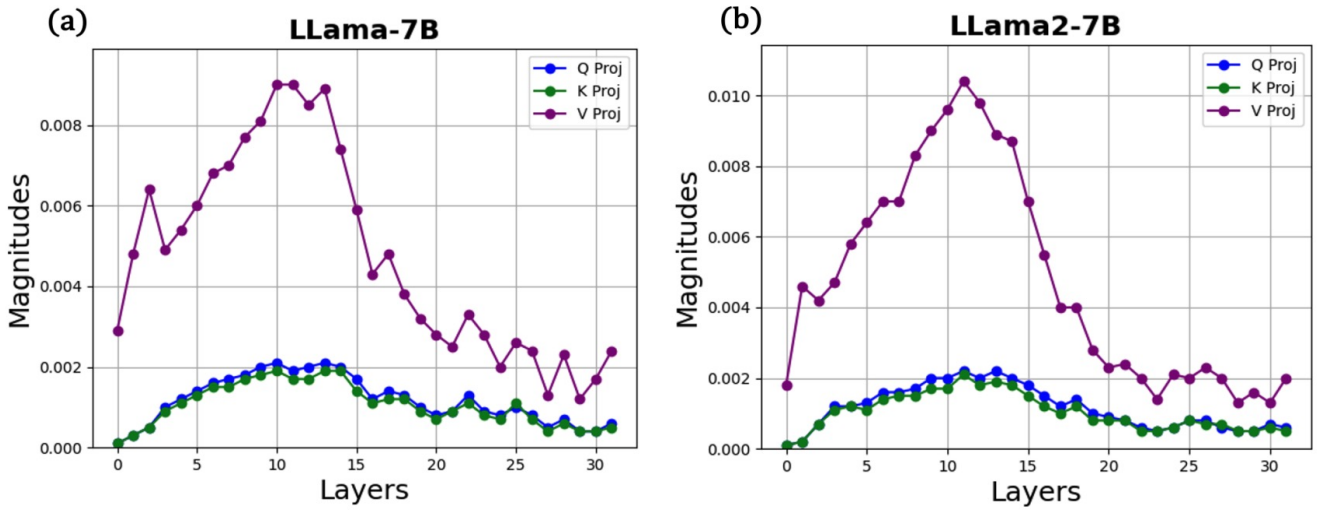


Figure 5. The magnitude of the gradient for the Q, K, and V vectors at each layer in LLMs.

$$\text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)$$

Experiments

- Accuracy comparison of LLaMA 7B, LLaMA 13B, LLaMA2 7B, and LLaMA3 8B with various PEFT methods on eight commonsense reasoning datasets.
- Bold texts dedicate the performance of SMT under the same numbers of parameters where LoRA, DoRA, and SpIEL achieve the best performance. Blue texts dedicate the best performance of SMT.
- The performance of LoRA, DoRA, and SpIEL under larger numbers of trainable parameters (next slide).

Model	PEFT method	#Params%	BoolQ	PIQA	SIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA	AVG
ChatGPT(175B)	-	-	73.1	85.4	68.5	78.5	66.1	89.8	79.9	74.8	77.0
LLaMA-7B	LoRA(Best)	0.83	67.5	80.8	78.2	83.4	80.4	78.0	62.6	79.1	76.3
	DoRA(Best)	0.84	69.7	83.4	78.6	87.2	81.0	81.9	66.2	79.2	78.4
	SpIEL(Best)	0.84	67.7	81.2	78.6	84.0	80.2	78.3	62.8	78.8	76.5
	SMT	0.84	68.7	81.7	78.3	91.6	78.8	84.1	68.7	77.4	**78.7**
	SMT(Best)	4.91	72.0	82.9	80.7	93.3	82.4	86.1	70.6	83.0	**81.4**
	Full Fine-tuning	100	69.9	84.2	78.9	92.3	83.3	86.6	72.8	83.4	81.4
LLaMA-13B	LoRA(Best)	0.67	72.1	83.5	80.5	90.5	83.7	82.8	68.3	82.4	80.5
	DoRA(Best)	0.68	72.4	84.9	81.5	92.4	84.2	84.2	69.6	82.8	81.5
	SpIEL(Best)	0.68	73.2	84.3	81.4	91.2	84.1	83.1	68.8	82.8	81.1
	SMT	0.68	71.1	84.4	81.7	93.7	83.2	86.7	73.7	85.2	**82.4**
	SMT(Best)	4.91	72.6	86.1	81.9	95.0	86.1	88.2	77.1	87.4	**84.3**
LLaMA2-7B	LoRA(Best)	0.83	69.8	79.9	79.5	83.6	82.6	79.8	64.7	81.0	77.6
	DoRA(Best)	0.42	72.0	83.1	79.9	89.1	83.0	84.5	71.0	81.2	80.5
	SpIEL(Best)	0.83	70.5	80.6	80.8	85.8	83.4	81.2	65.8	81.8	78.3
	SMT	0.84	72.0	83.8	80.8	93.3	82.8	86.7	74.0	81.0	**81.8**
	SMT(Best)	4.91	72.6	85.2	82.0	94.4	85.7	87.8	74.5	85.0	**83.4**
	Full Fine-tuning	100	72.8	83.4	78.7	92.7	85.5	86.2	74.7	83.4	82.2
LLaMA3-8B	LoRA(Best)	0.70	70.8	85.2	79.9	91.7	84.3	84.2	71.2	79.0	80.8
	DoRA(Best)	0.71	74.6	89.3	79.9	95.5	85.6	90.5	80.4	85.8	85.2
	SpIEL(Best)	0.70	72.1	83.6	80.0	91.8	85.4	91.2	76.8	80.8	82.7
	SMT	0.71	75.7	88.4	81.4	96.2	88.2	92.7	83.2	88.6	**86.8**
	SMT(Best))	3.01	75.1	89.9	82.4	96.3	88.8	92.6	82.8	89.6	**87.2**

Model	PEFT method	#Params%	GSM8k	SingleEq	SVAMP	MultiArith	AddSub	AQuA	AVG
LLaMA-7B	LoRA(Best)	0.86	35.4	83.2	52.1	92.8	83.4	18.6	60.9
	DoRA(Best)	0.86	35.2	83.7	51.8	92.8	82.8	20.2	61.1
	SMT	0.86	34.2	84.6	53.6	91.5	85.8	23.6	62.2
	SMT(Best)	1.26	35.6	85.3	54.8	93.4	86.8	24.2	63.4

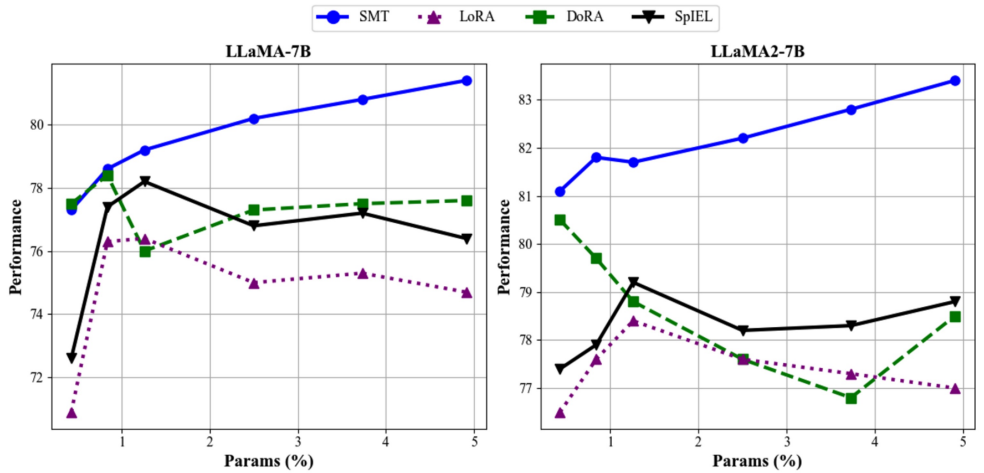
- SMT, LoRA and DoRA reproduction, and experiment results on Math10K dataset.

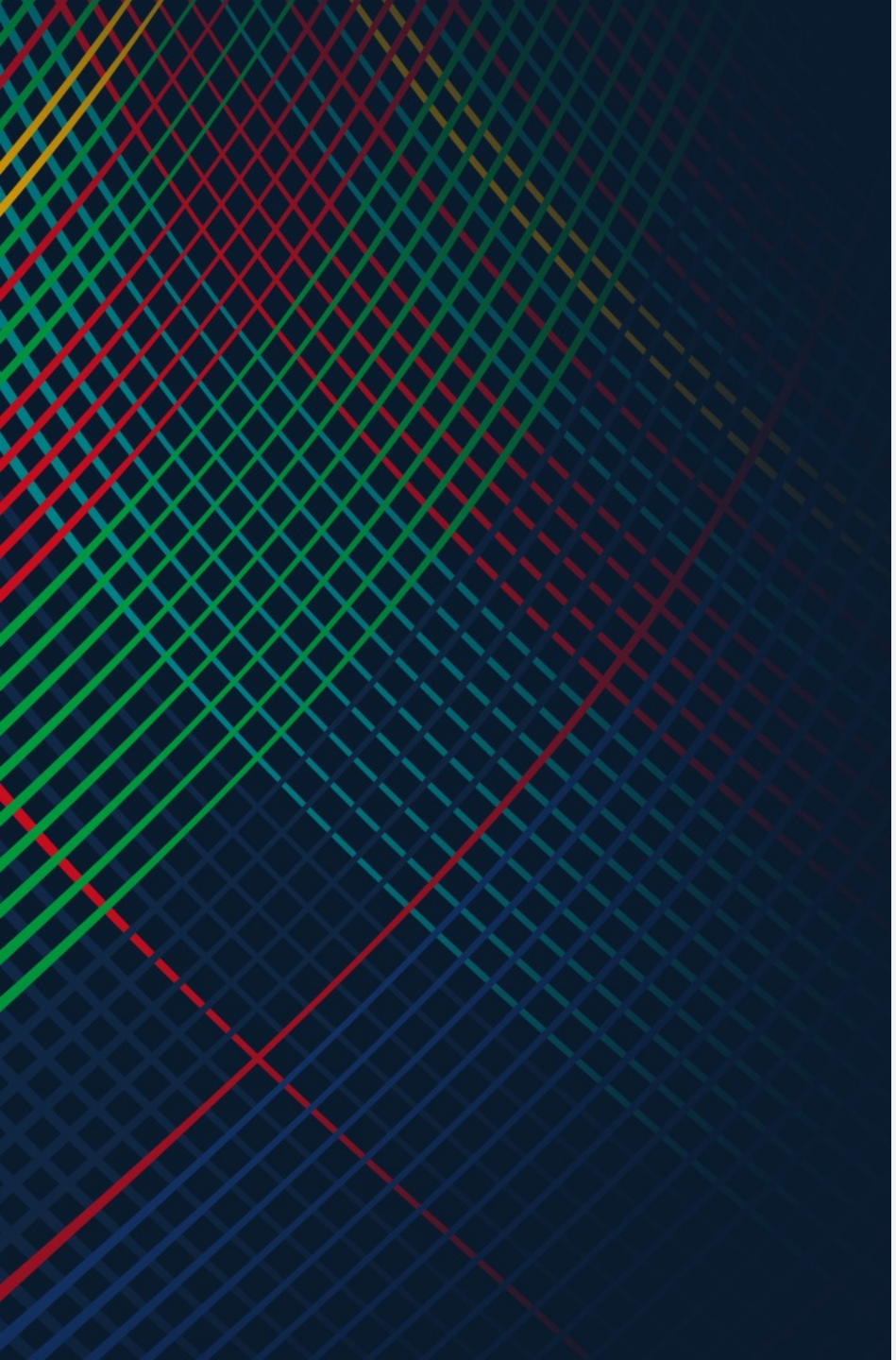
Plateau Issue

- We scale up the model size and presents how the performance of LoRA, DoRA, and SpIEL will be under larger number of trainable parameters.

Table 3: Accuracy comparison of LoRA, DoRA, and SMT under different scaling of trainable parameters on `commonsense` datasets. Given certain base model and PEFT method, we gradually increase the number of trainable parameters on each line from left to right. On each line, the best performing model has *.

	Method	0.43	0.84	1.26	2.50	3.73	4.91
LLaMA-7B	**LoRA**	70.9	**76.3***	76.4	75.0	75.3	74.7
	SpIEL	72.6	77.4	**78.2***	76.8	77.2	76.4
	DoRA	77.5	**78.4***	76.0	77.3	77.5	77.6
	SMT	77.3	78.6	79.2	80.2	80.8	**81.4***
LLaMA2-7B	**LoRA**	76.5	77.6	**78.4***	77.6	77.3	77.0
	SpIEL	77.4	77.9	**79.2***	78.2	78.3	78.8
	DoRA	**80.5***	79.7	78.8	77.6	76.8	78.5
	SMT	81.1	81.8	81.7	82.2	82.8	**83.4***





Thank You

Xuan Jiang

xuanj@mit.edu

More Details…

- Theoretical Supports
- System Implementation Details
- MLP versus Attentions, which is more crucial?