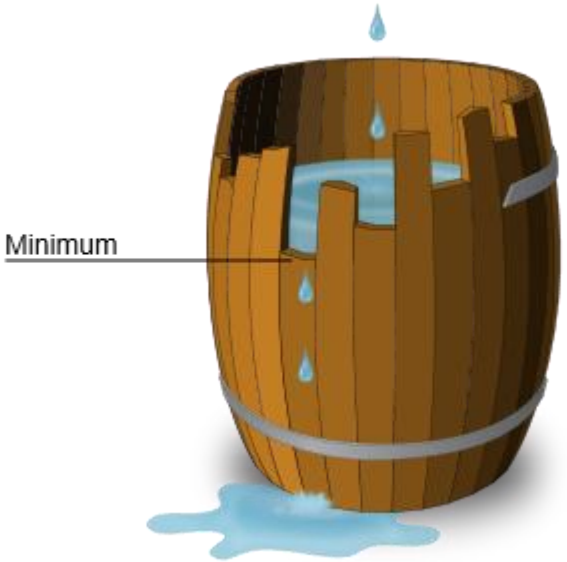
Pay attention to small weights

Chao Zhou, Advait Gadhikar, Tom Jacobs, Rebekka Burkholz

NeurIPS 2025



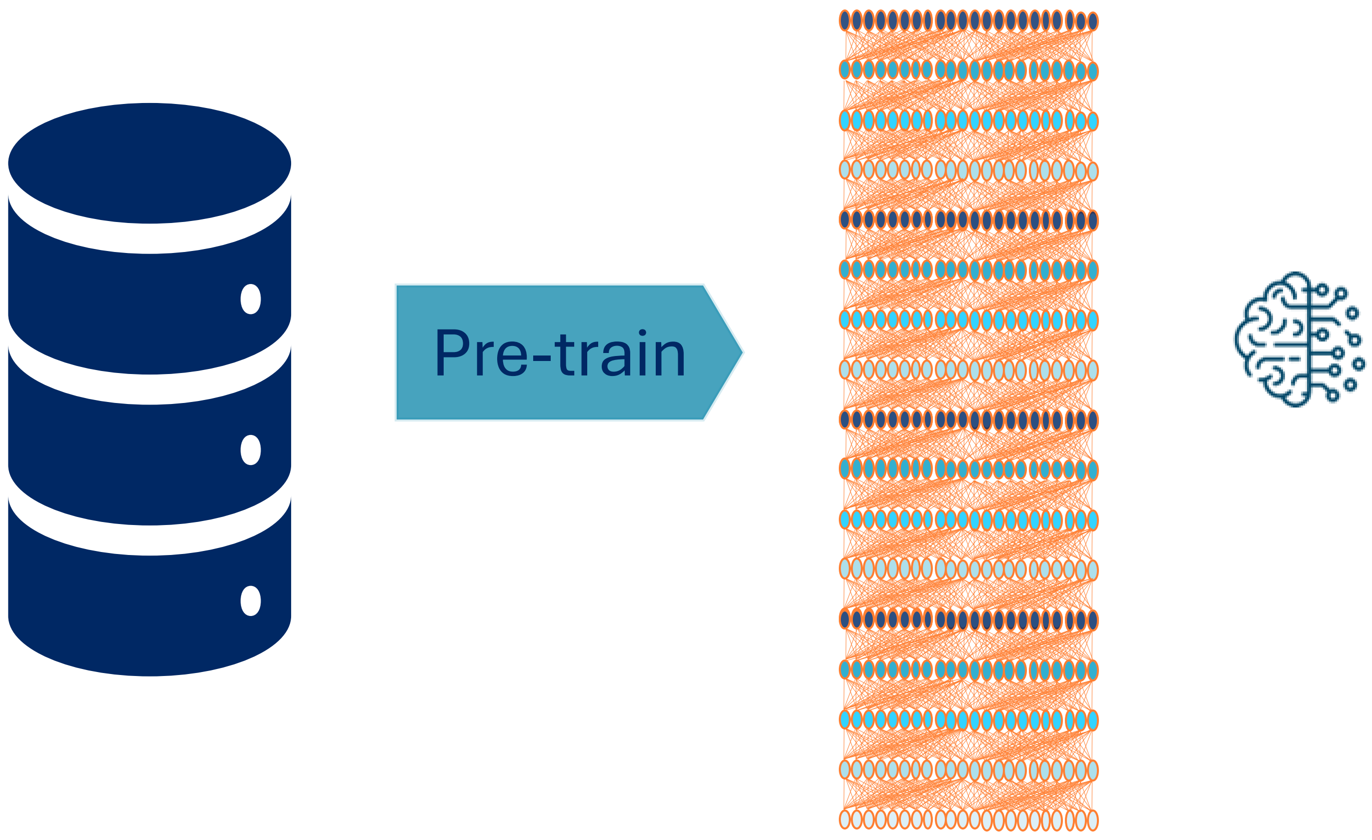
Outline

- Intro to finetuning
- Parameter-efficient finetuning
- Method
- Conclusion

Outline

- Intro to finetuning
- Parameter-efficient finetuning
- Method
- Conclusion

Intro to finetuning- large models



ChatGPT

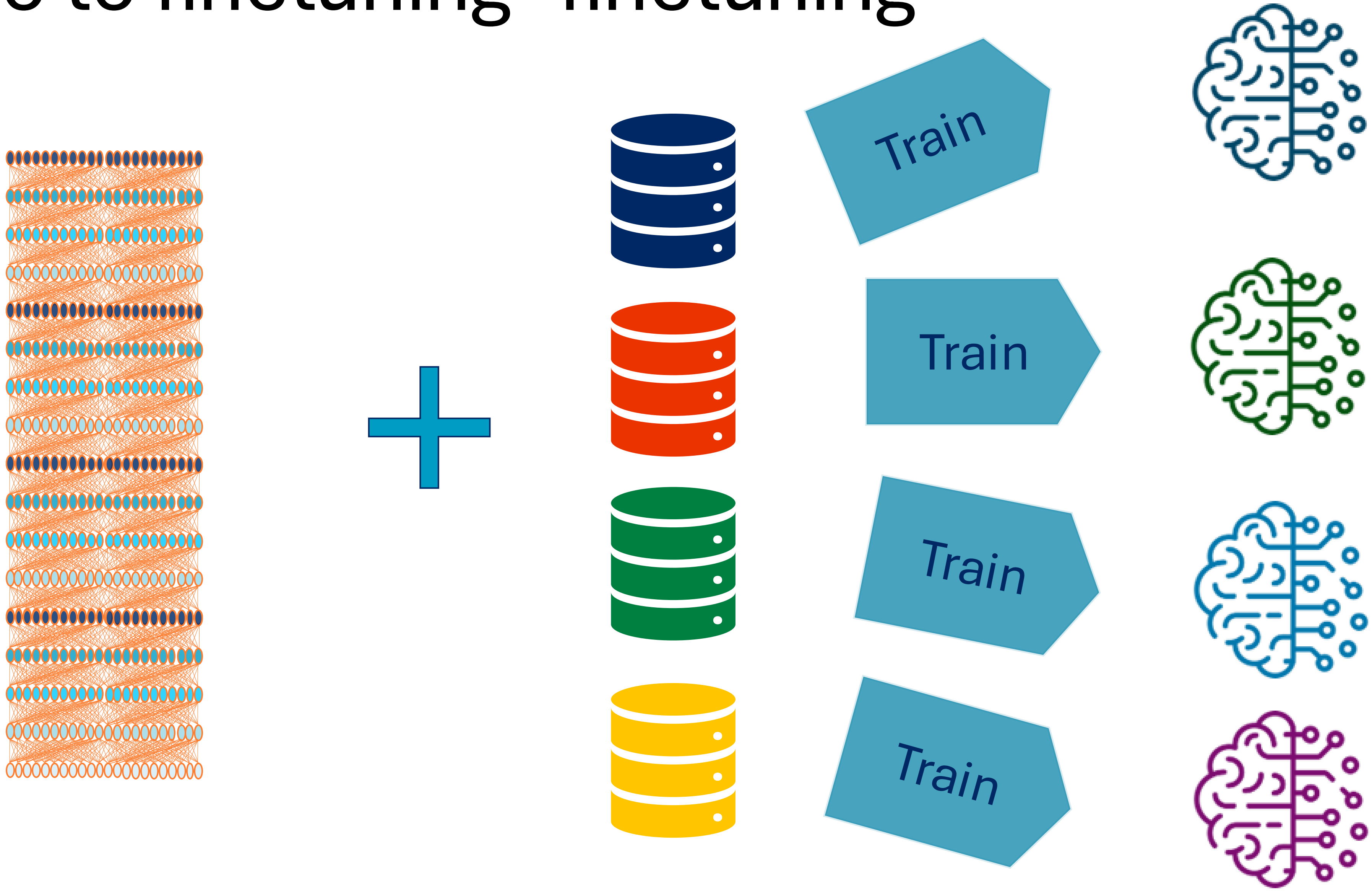
Gemini

Qwen

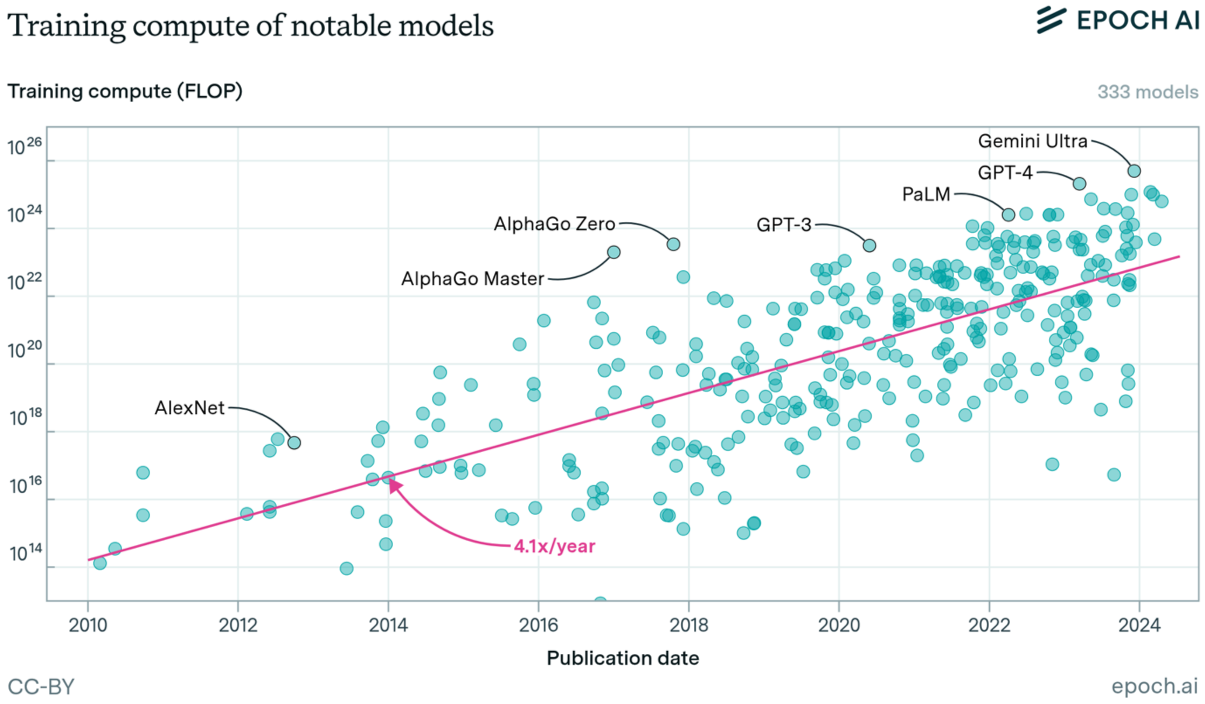
deepseek

...

Intro to finetuning- finetuning

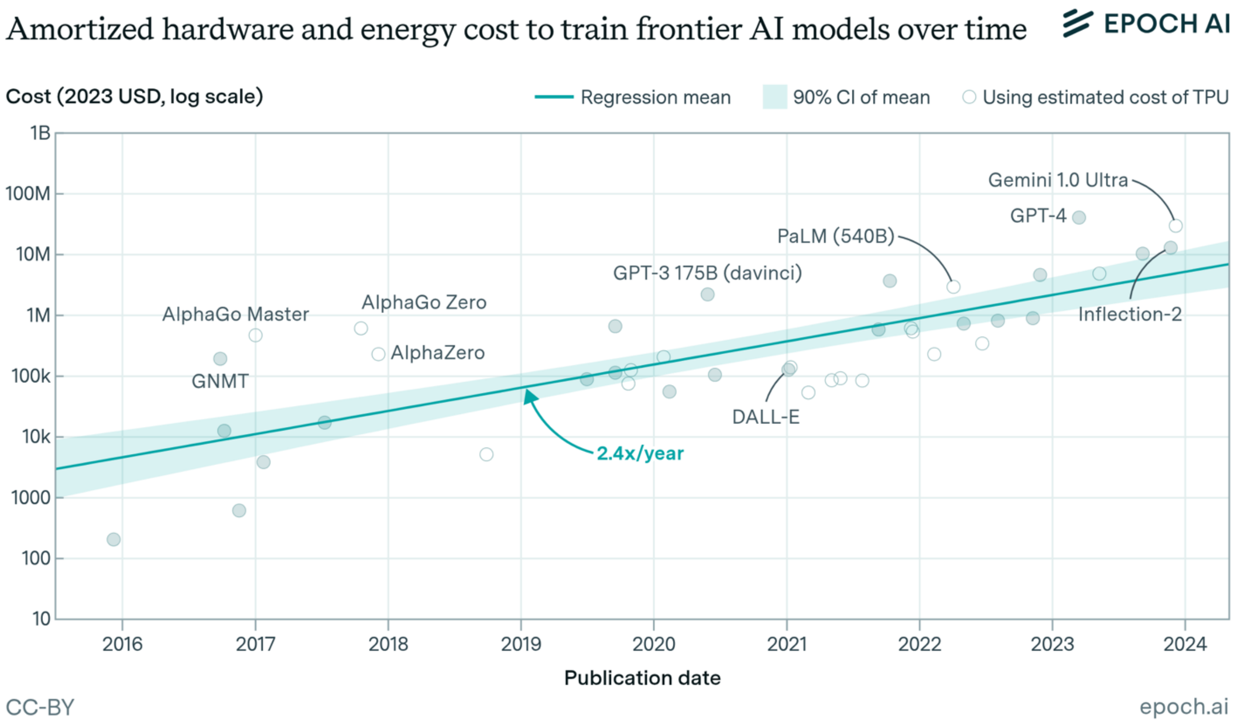


Intro to finetuning-but what is the cost?



compute

Intro to finetuning-but what is the cost?



money

Outline

- Intro to finetuning
- Parameter-efficient finetuning
- Method
- Conclusion

Parameter-efficient finetuning

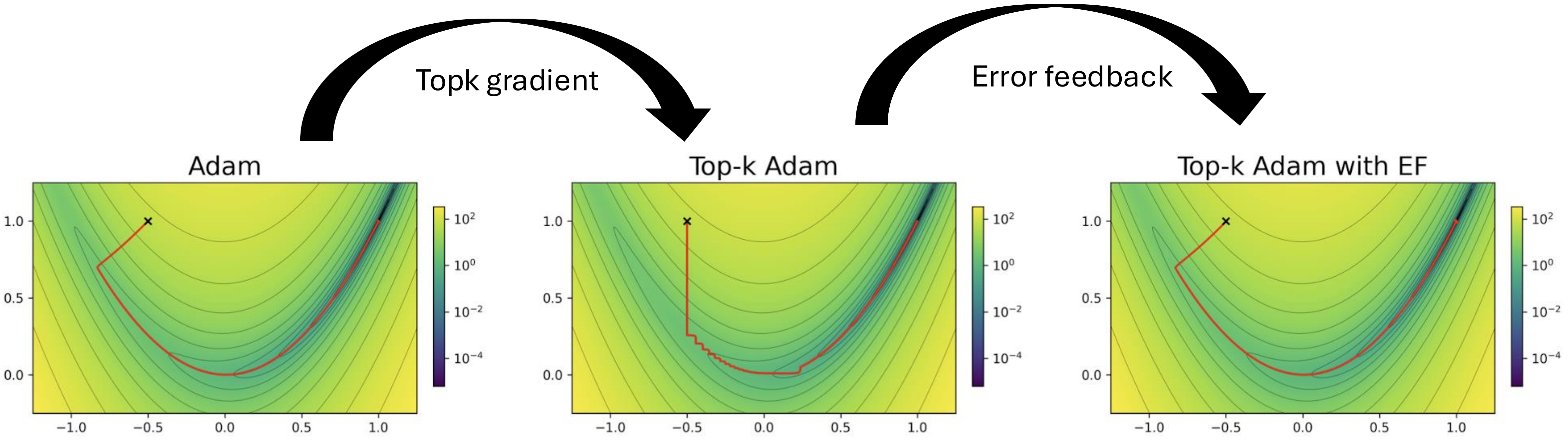
- Tech to reduce FT cost, including
  - ✓ identify subset of param to update: LoRA
  - ✓Quantise optimizer state: Adamw-8bit
  - ✓Project gradient to subspace: GaLore
  - ✓…

Outline

- Intro to finetuning
- Parameter-efficient finetuning
- Method
- Conclusion

Method

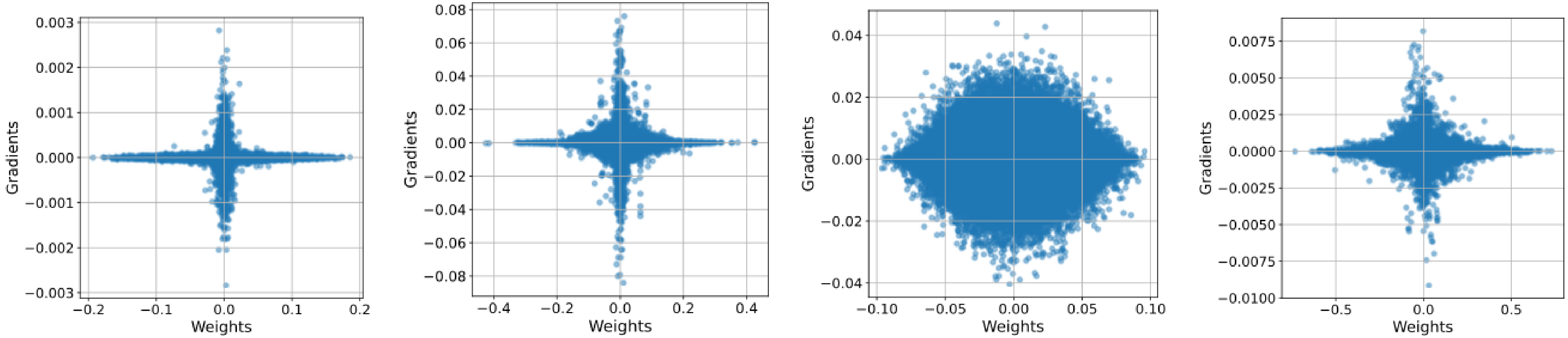
From optimization perspective



Known as MicroAdam

Method

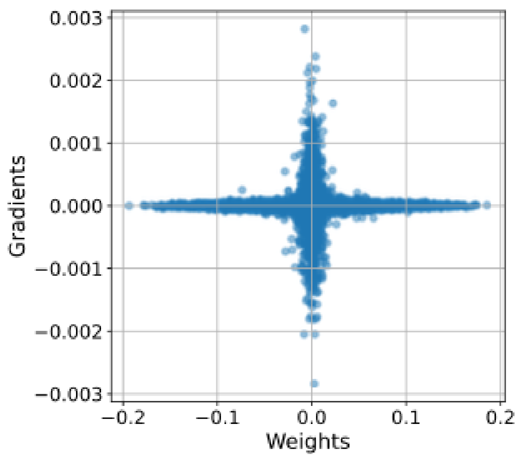
Let’s examine magnitude vs gradient



(a) NLP task: FT. (b) CV task: FT. (c) CV task: Train from scratch at step=0. (d) CV task: Train from scratch at step=78100.

Method

Let’s examine magnitude vs gradient



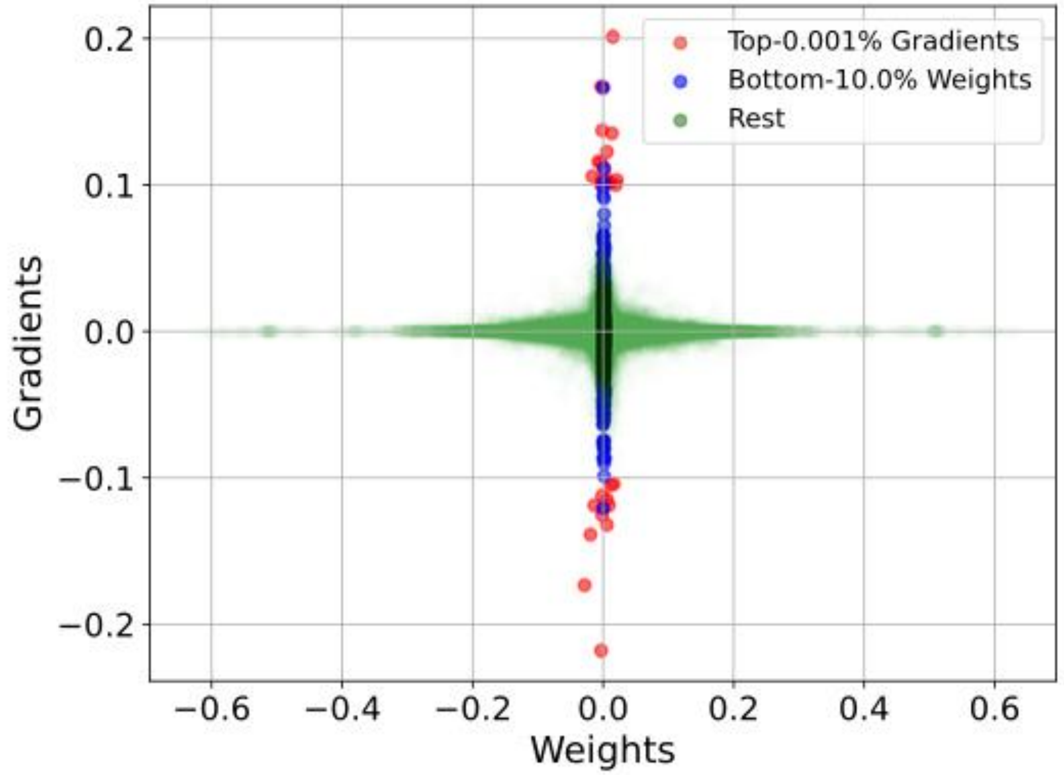
✓ large gradient -> small weights;

✓ stronger correlation in FT than pre-train;

Q: are they the same?

Method

Overlap between small weights and large gradients

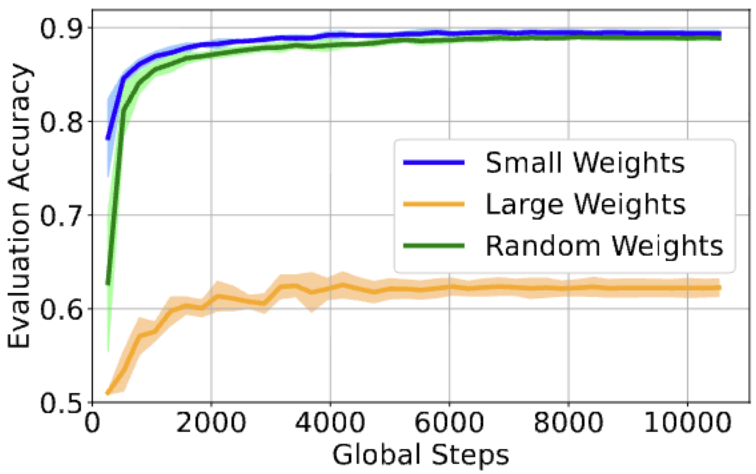


Weights and gradients are not identical.

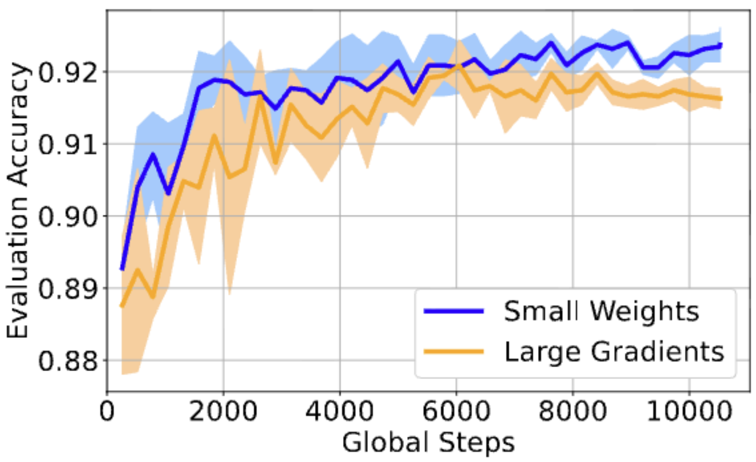
Can we update only small weights?

Method

Ablation study



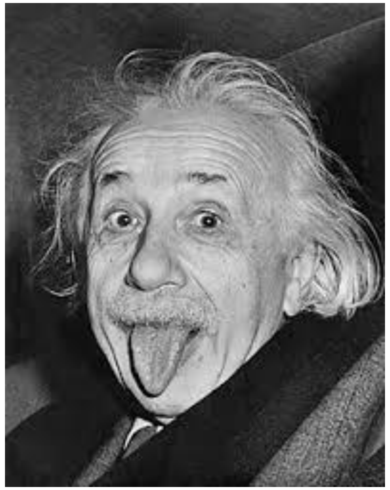
(a) Small vs. large vs. random weights.



(b) Small weights vs. large gradients.

Method

Intuition behind



Pretrain large weights = critical features

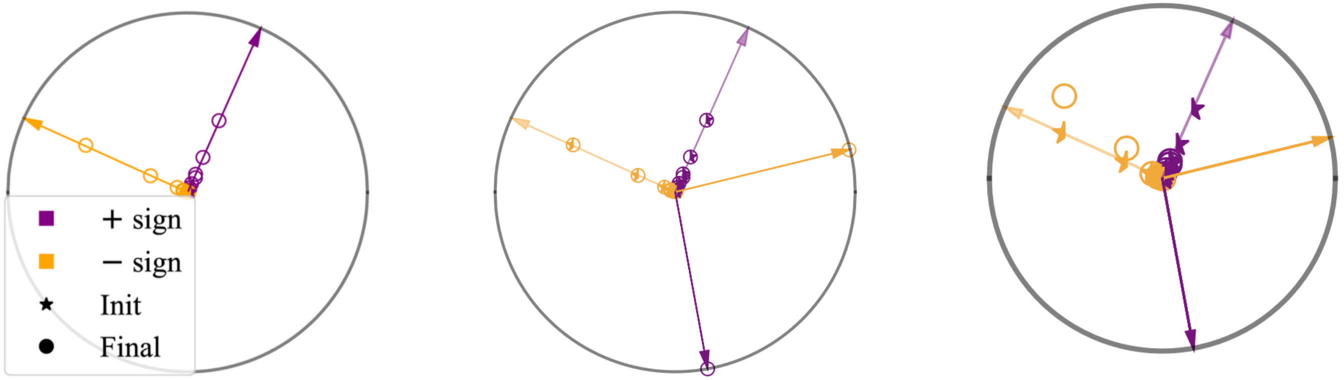
Knowledgeable but stubborn

Small weights = plasticity

”Less” Knowledgeable but open

Method

Theoretical motivation

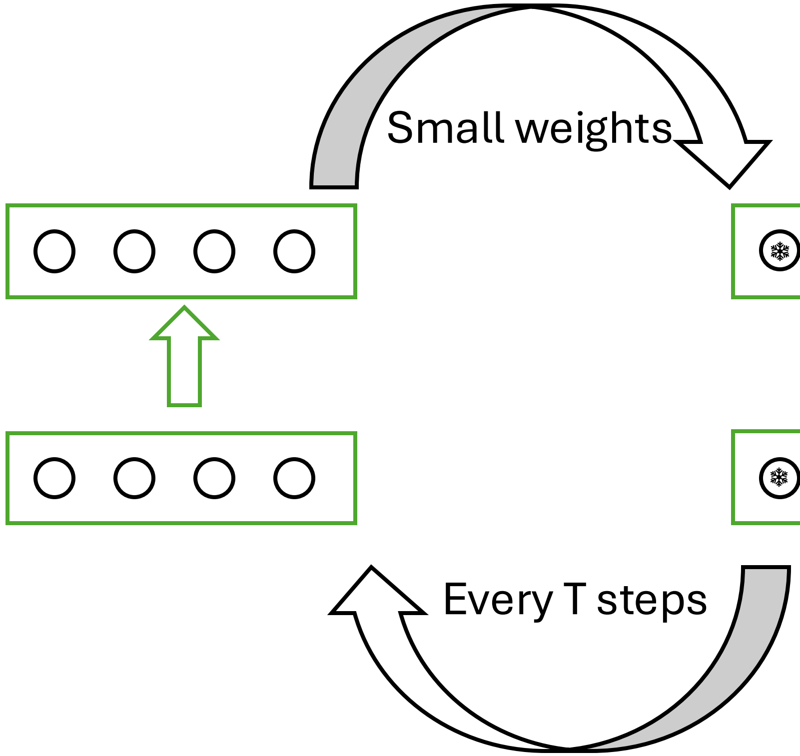


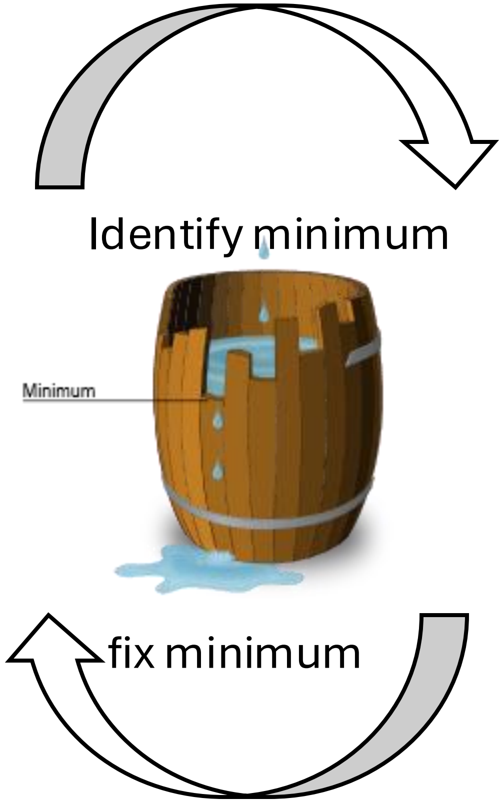
(a) Dense feature learning. (b) Nano GD. (c) Largest gradients.

Figure 7: nano gradient descent provably prevents catastrophic forgetting. (a) Two layer student network learns teach networks representation. (b) Nano gradient descent keeps the original representation while learning the two extra neuron. (c) The largest gradients can lead to learning completely different representations when the task transferability is high.

Method

- NanoAdam: an opt for finetuning





Method

- Exp on FT Llama 3.2 3B on Commonsense benckmark

Method	HellaSwag	Winogrande	PIQA	ARC-Easy	ARC-Challenge	OpenBookQA	SocialIQA	BoolQ	AVG.	memory	time (h)
MicroAdam	92.44	83.27	85.91	85.82	73.38	81.40	79.07	63.94	80.65	28.64	12.25
NanoAdam	93.21	82.48	86.07	85.86	74.91	82.20	79.99	71.71	**82.05**	**25.30**	**10.11**
AdamW	77.71	74.11	63.33	83.63	68.34	75.20	76.10	66.67	73.14	37.75	21.8

Method

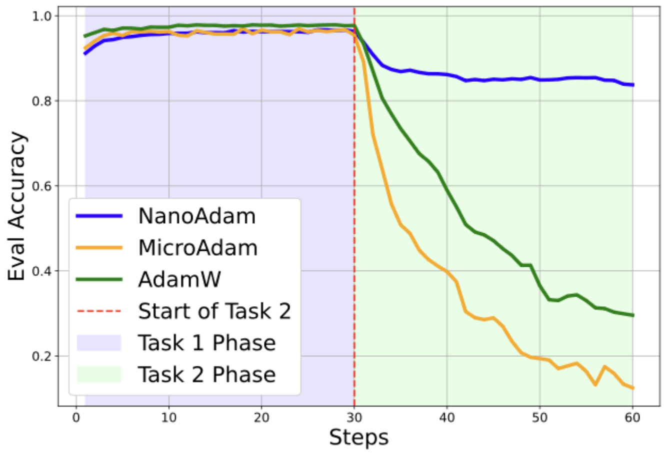
- Avoid forgetting

Averaged evaluation accuracy and parameter change in ℓ_2 distance, alongside learning rate.

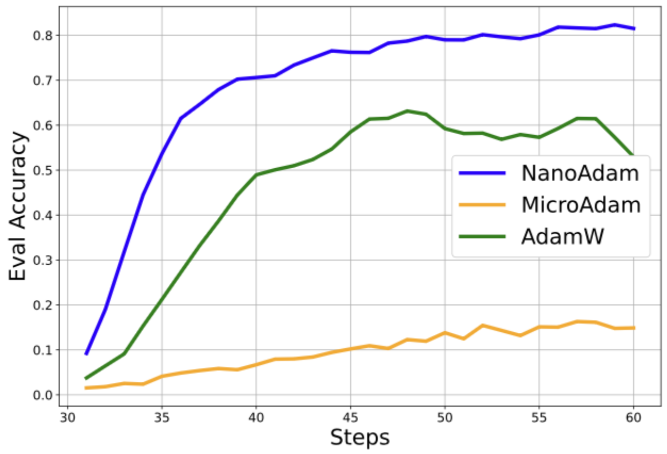
Algorithm	**LR (task1)**	**LR (task2)**	**AVG. Acc**	$\boldsymbol{\ell_2}$ **Distance**
AdamW	1e−4	1e−4	96.81%	0.83
MicroAdam	1e−4	1e−3	95.23%	0.75
NanoAdam	1e−3	2e−3	**98.95%**	**0.68**

Method

- Avoid forgetting



(a) Generalization on Task 1.



(b) Generalization on Task 2.

Outline

- Intro to finetuning
- Parameter-efficient finetuning
- Method
- Conclusion

Conclusion

Small weight is sufficient to finetuning task;

- Implicit weight decay;
- Better generalization;
- Save memory;
- Avoid catastrophic forgetting.