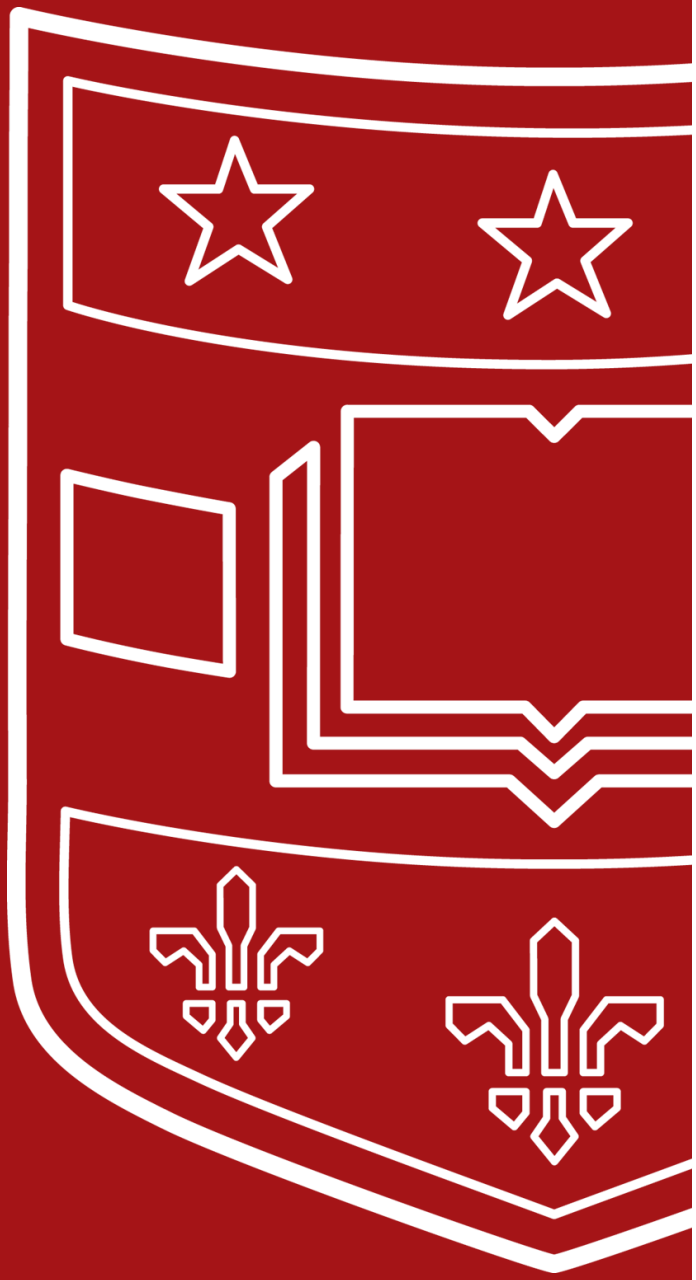
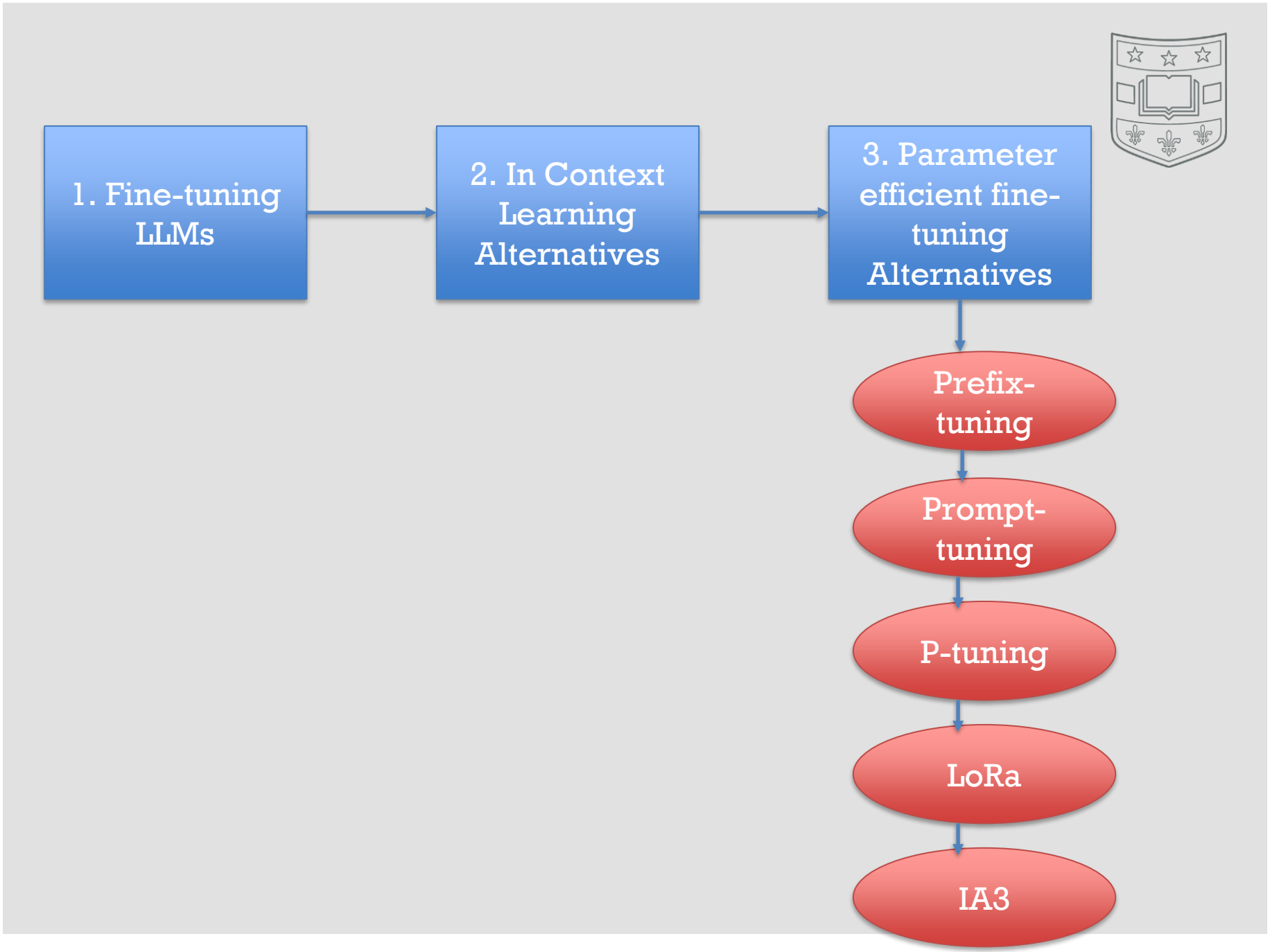
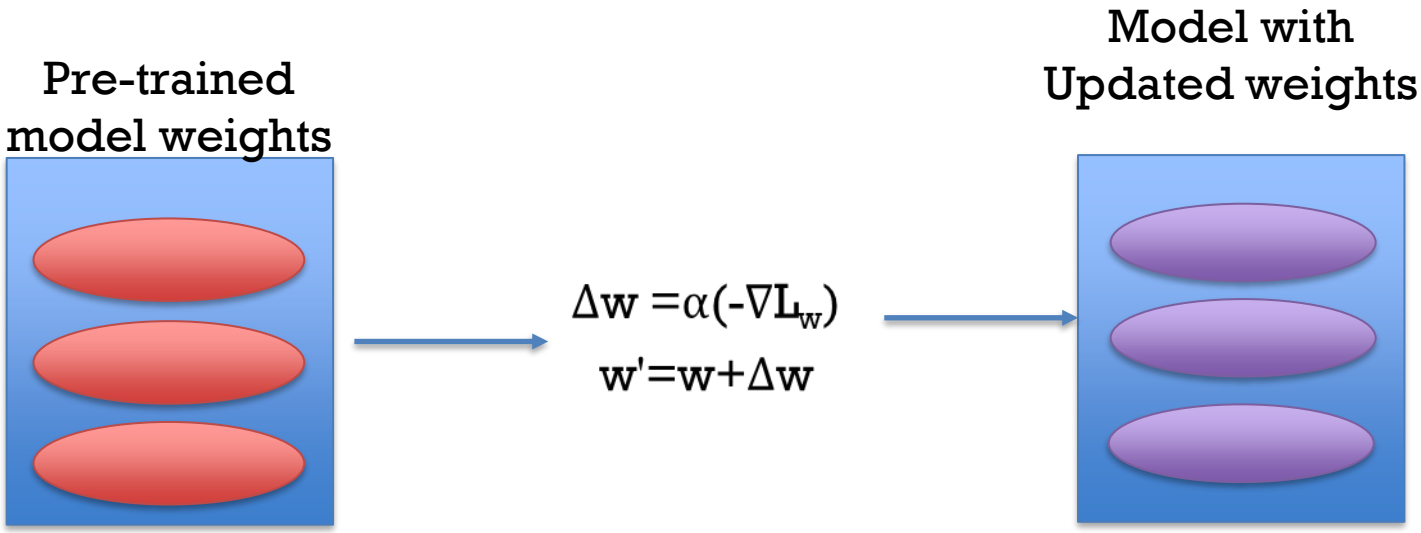
Parameter efficient fine-tuning

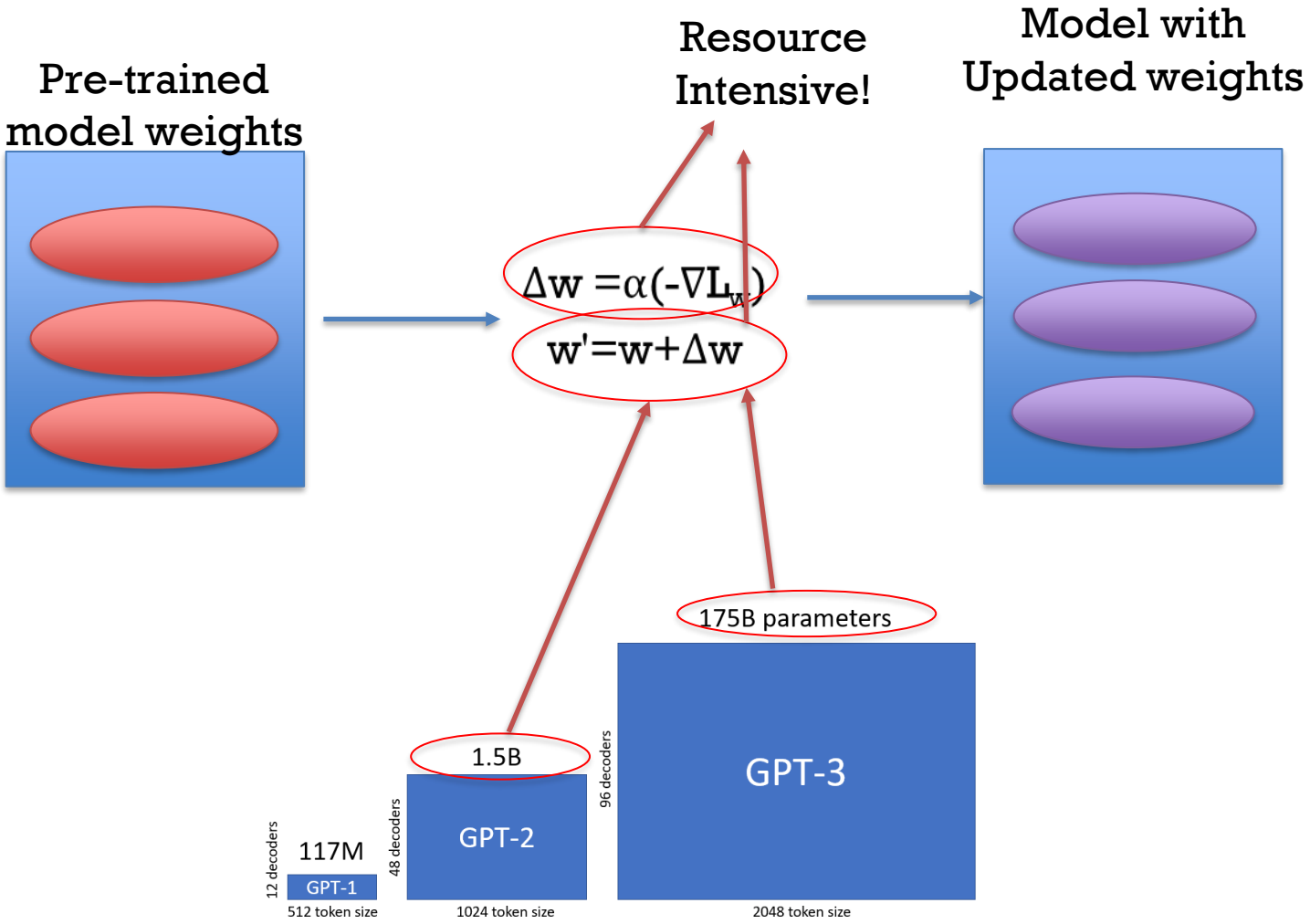



1. Fine-tuning LLMs
2. In Context Learning Alternatives
3. Parameter efficient fine-tuning Alternatives
Prefix-tuning
Prompt-tuning
P-tuning
LoRa
IA3

Fine-tuning LLMs (or any deep learning model)



Problem 1: As models get larger, fine-tuning becomes more computationally intensive!

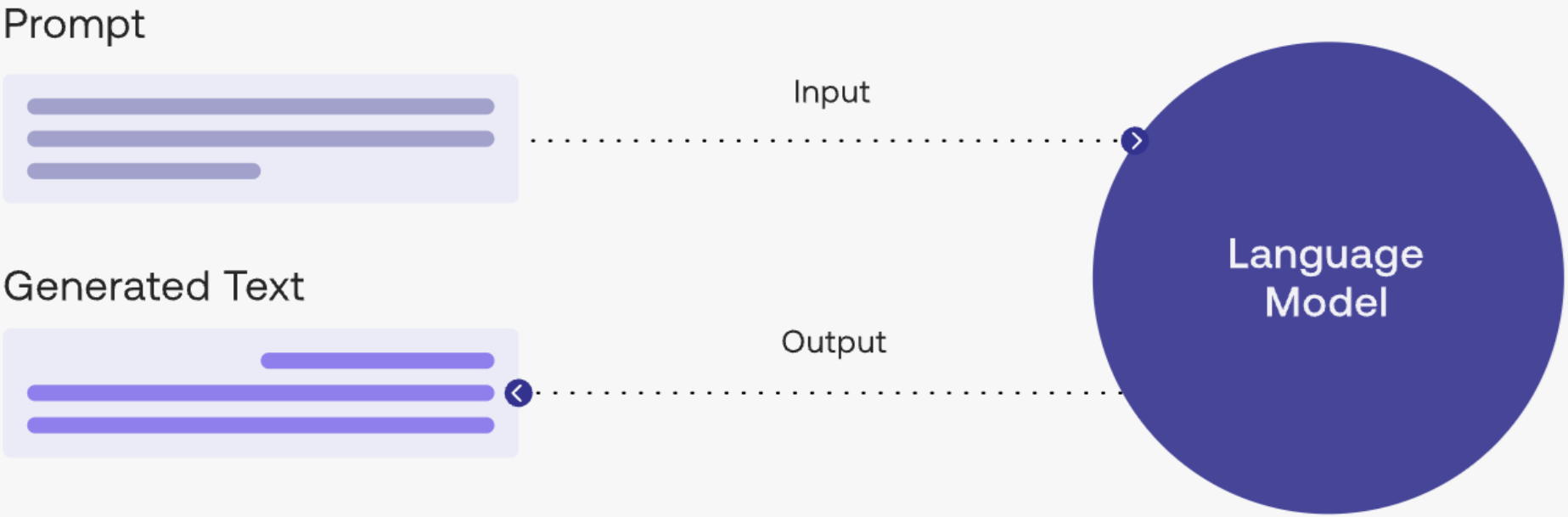


Other problems

1. Specialized for a single domain
   - Have to fine-tune for each specific domain
2. Require sufficiently **abundant** and **representative** data
   - Labelling can be **expensive**!!!

In context learning

Idea: Feed it input-target examples (“shots”)



In context learning (test)

Example of a 4-shot input:

{Please unscramble the letters into a word, and write that word:

- asinoc=casino,
- yfrogg=froggy,
- plesim=simple,
- iggestb=biggest,
- astedro=?}

In context learning (test)

Example of a 4-shot input:

{Please unscramble the letters into a word, and write that word:

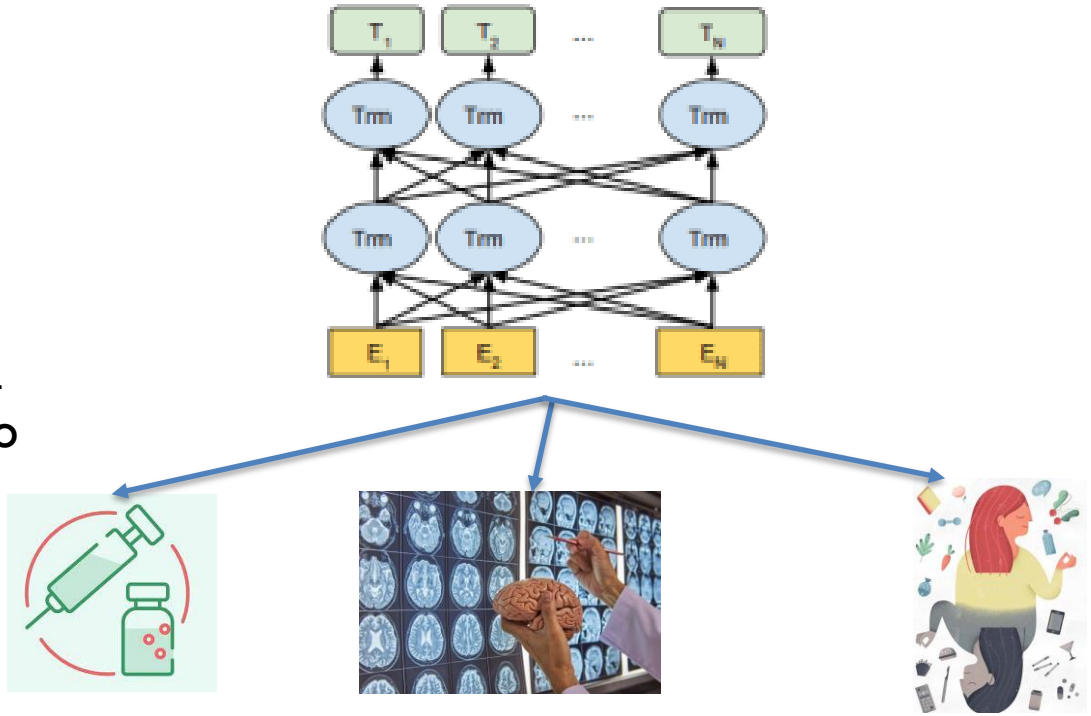
- asinoc=casino,
- yfrogg=froggy,
- plesim=simple,
- iggestb=biggest,
*- astedro=**roasted**}*

Advantages to ICL

1. Enables single model to perform many task immediately without fine-tuning

2. Allows for mixed task batches

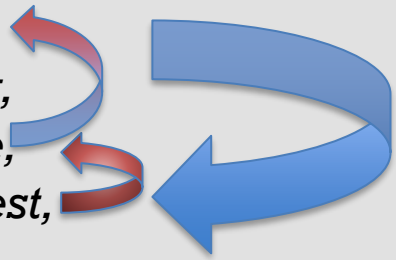
- different examples in a batch of data correspond to different task.



Problems with ICL

Can display unintuitive behavior.

{Please unscramble the letters into a word, and write that word:
- asinoc=casino,
- yfrogg=froggy,
- plesim=simple,
- iggestb=biggest,
- astedro=?}



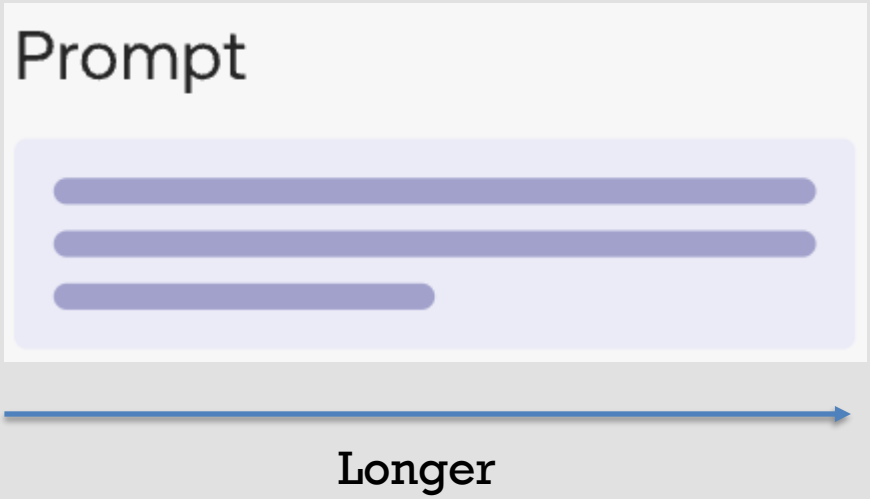
~~*Roasted*~~

Is it really learning???

Calibrate Before Use:
Improving Few-Shot Performance of Language Models

Tony Z. Zhao [*1] Eric Wallace [*1] Shi Feng [2] Dan Klein [1] Sameer Singh [3]

Problems with ICL

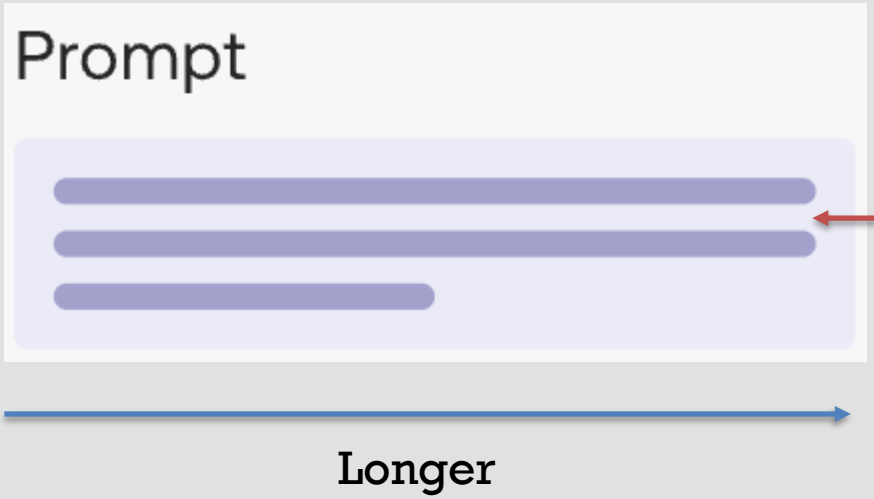


=

More expensive

&

More Memory



Limited by max token length inputs


1. Fine-tuning LLMs
2. In Context Learning Alternatives
3. Parameter efficient fine-tuning Alternatives

Welcome to the world of parameter efficient fine-tuning

What is parameter efficient fine-tuning? Definition by Li & Liang (2021)

Lightweight fine-tuning

Freezes most of the pretrained parameters and modifies the pretrained model with small trainable modules

Goal: identify high-performing architectures of the modules and the subset of pretrained parameters to tune

Definition by Liu et al (2022)

Provides **strong** performance on any task

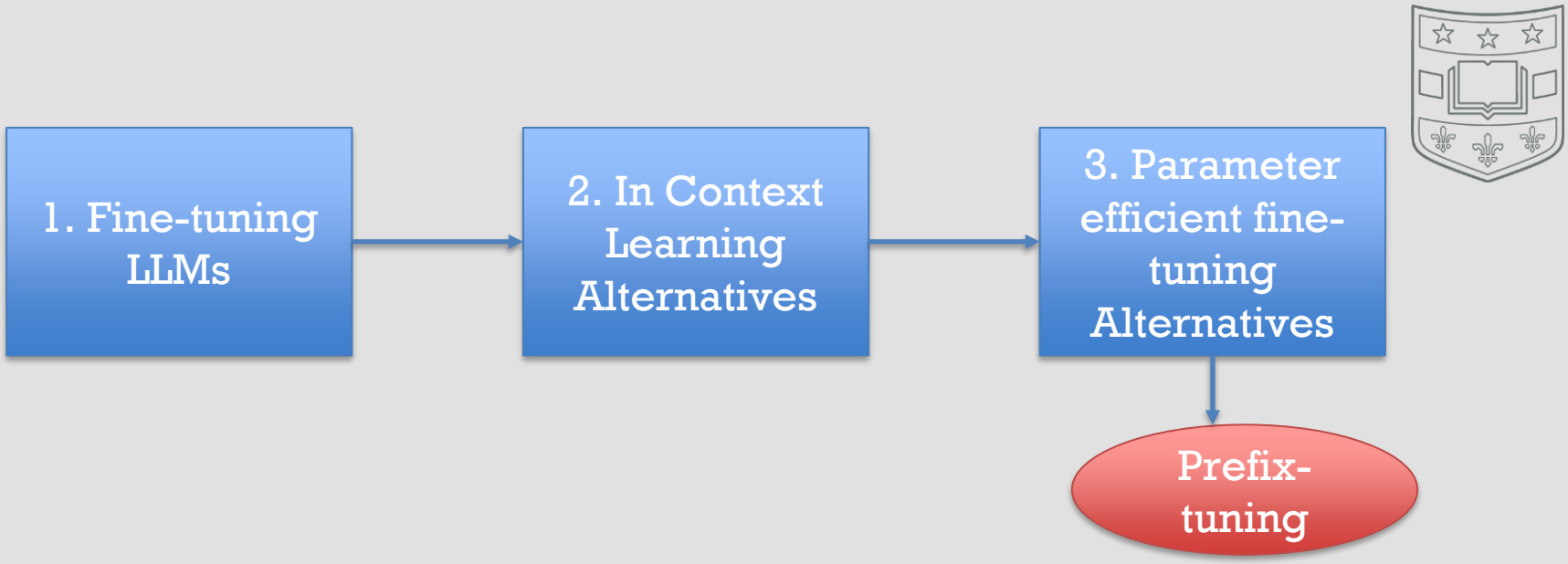
Without manual tuning or per-task adjustments

Realistic option in few-shot settings where limited labeled data is available.

How?

Exploiting tricks that allow the model to be tuned but adjusting only a **small amount of parameters**

While keeping the model's remaining weights **frozen**


1. Fine-tuning LLMs
2. In Context Learning Alternatives
3. Parameter efficient fine-tuning Alternatives
Prefix-tuning

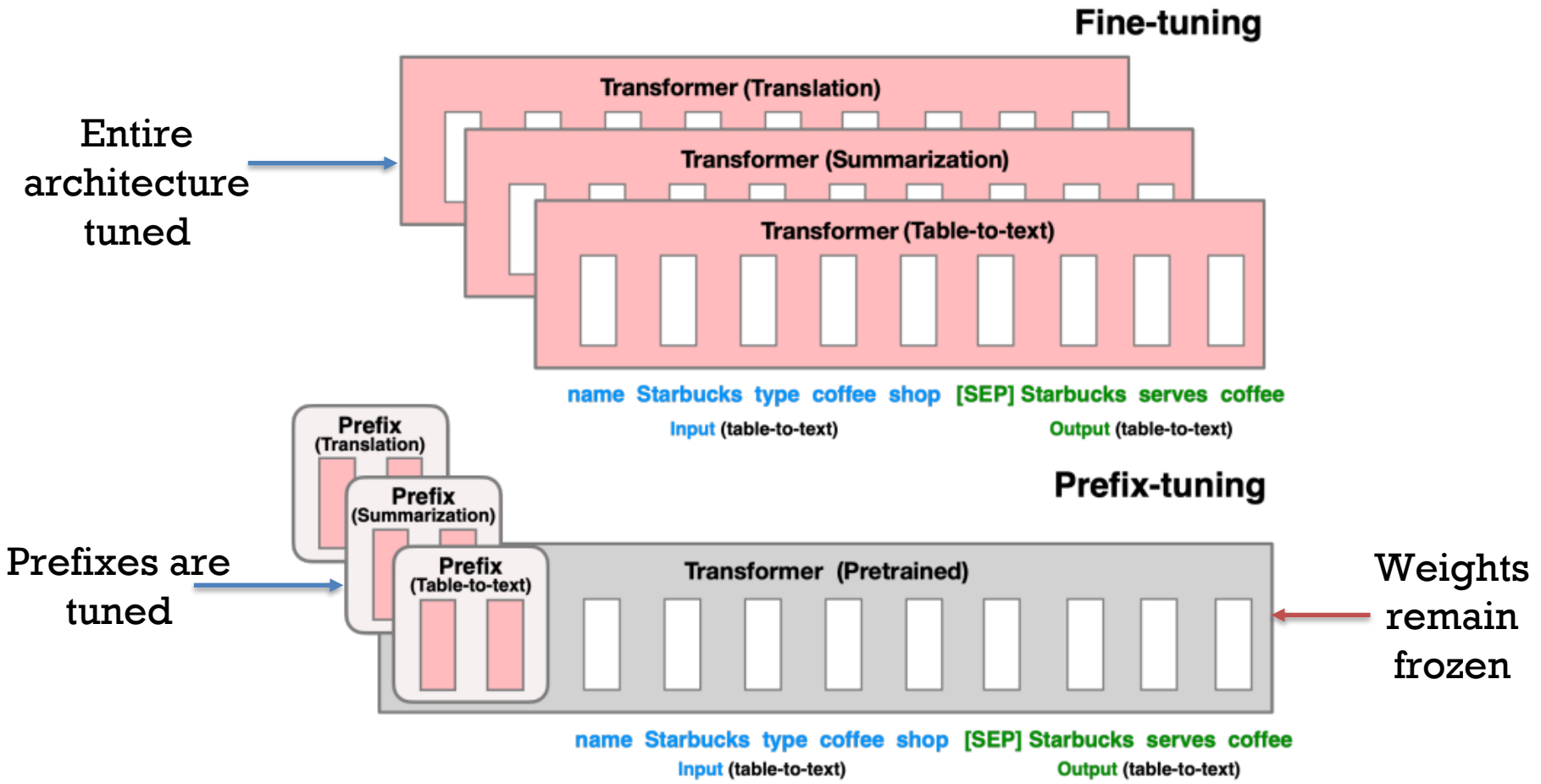
Prefix-Tuning: Optimizing Continuous Prompts for Generation

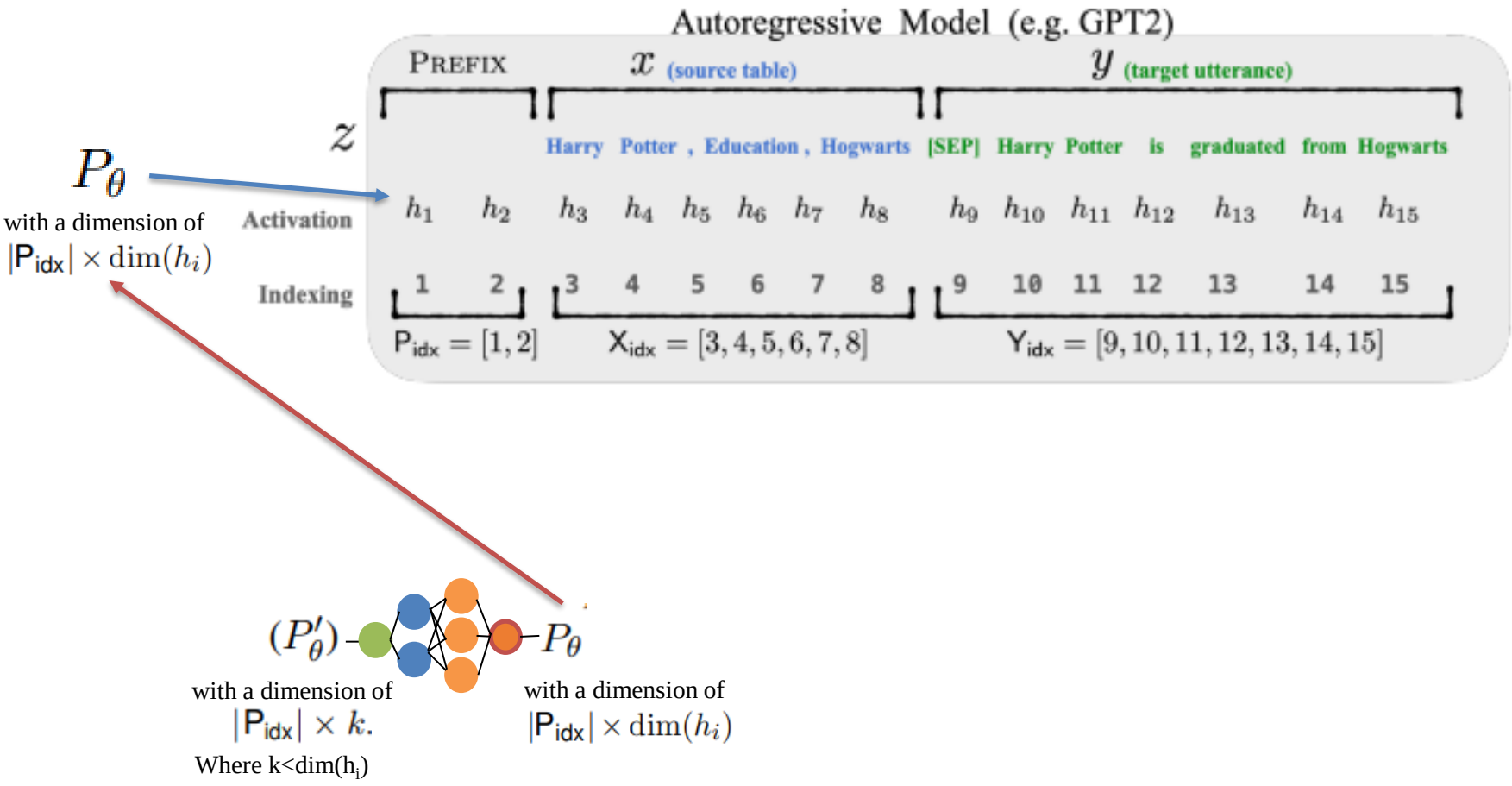
Xiang Lisa Li, Percy Liang

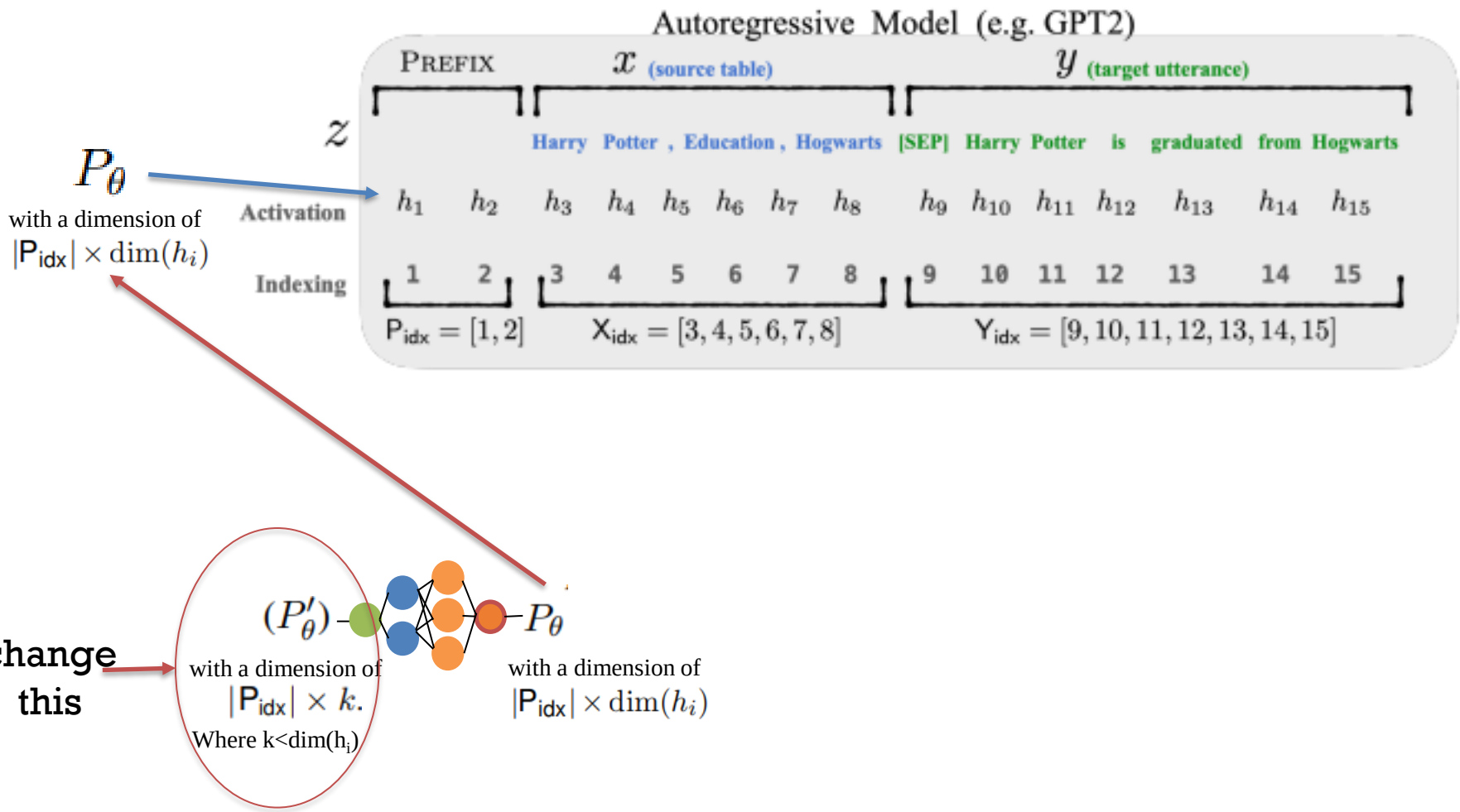
Abstract

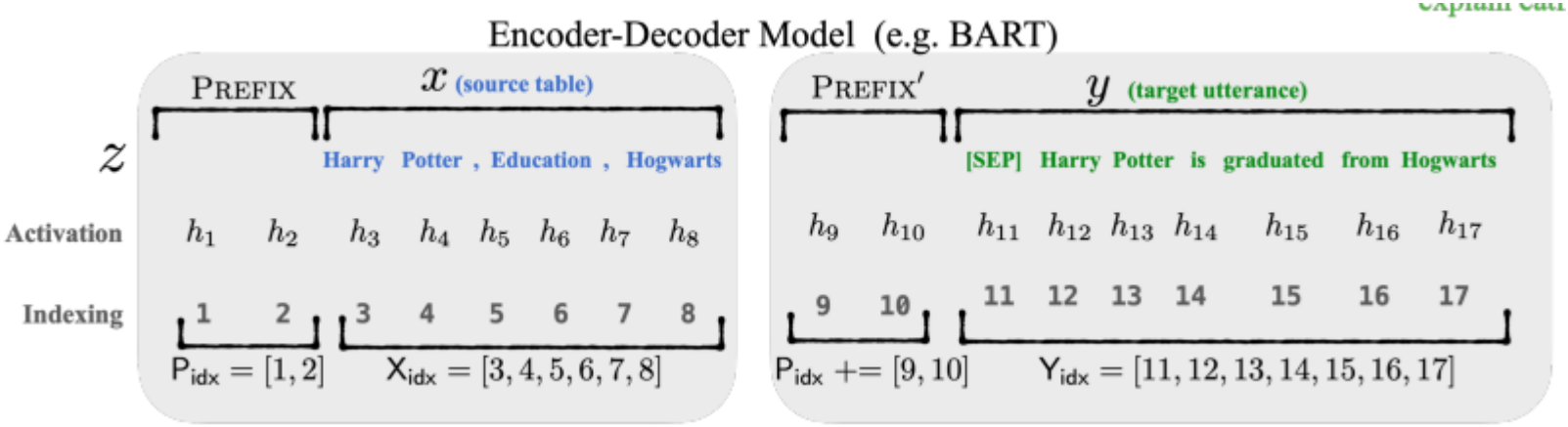
Fine-tuning is the de facto way of leveraging large pretrained language models for downstream tasks. However, fine-tuning modifies all the language model parameters and therefore necessitates storing a full copy for each task. In this paper, we propose prefix-tuning, a lightweight alternative to fine-tuning for natural language generation tasks, which keeps language model parameters frozen and instead optimizes a sequence of continuous task-specific vectors, which we call the prefix. Prefix-tuning draws inspiration from prompting for language models, allowing subsequent tokens to attend to this prefix as if it were "virtual tokens". We apply prefix-tuning to GPT-2 for table-to-text generation and to BART for summarization. We show that by learning only 0.1% of the parameters, prefix-tuning obtains comparable performance in the full data setting, outperforms fine-tuning in low-data settings, and extrapolates better to examples with topics that are unseen during training.

Prefix Tuning

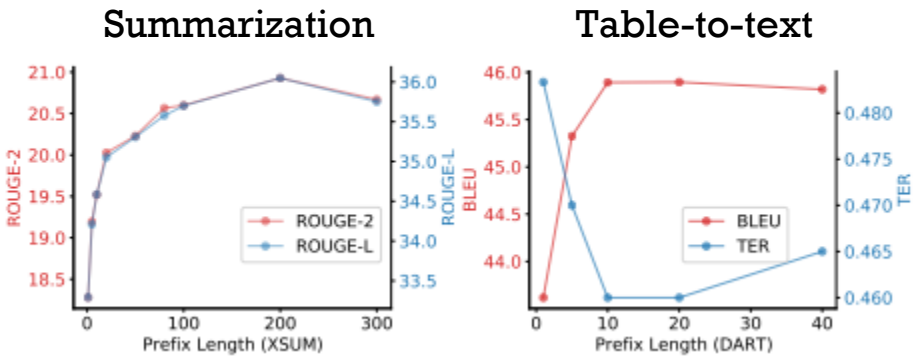



Autoregressive Model (e.g. GPT2)
PREFIX
x (source table)
y (target utterance)
z
Harry Potter , Education , Hogwarts [SEP] Harry Potter is graduated from Hogwarts
P_θ
with a dimension of
$|\mathsf{P}_{\mathsf{idx}}| \times \dim(h_i)$
Activation
h_1 h_2 h_3 h_4 h_5 h_6 h_7 h_8 h_9 h_{10} h_{11} h_{12} h_{13} h_{14} h_{15}
Indexing
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
$\mathsf{P}_{\mathsf{idx}} = [1, 2]$
$\mathsf{X}_{\mathsf{idx}} = [3, 4, 5, 6, 7, 8]$
$\mathsf{Y}_{\mathsf{idx}} = [9, 10, 11, 12, 13, 14, 15]$
(P'_θ)
P_θ
with a dimension of
$|\mathsf{P}_{\mathsf{idx}}| \times k.$
Where k<dim(h_i)
with a dimension of
$|\mathsf{P}_{\mathsf{idx}}| \times \dim(h_i)$


Autoregressive Model (e.g. GPT2)
PREFIX
x (source table)
y (target utterance)
z
Harry Potter , Education , Hogwarts [SEP] Harry Potter is graduated from Hogwarts
P_θ
with a dimension of
$|P_{idx}| \times \dim(h_i)$
Activation
h_1 h_2 h_3 h_4 h_5 h_6 h_7 h_8 h_9 h_{10} h_{11} h_{12} h_{13} h_{14} h_{15}
Indexing
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
$P_{idx} = [1, 2]$
$X_{idx} = [3, 4, 5, 6, 7, 8]$
$Y_{idx} = [9, 10, 11, 12, 13, 14, 15]$
change this
(P'_θ)
with a dimension of
$|P_{idx}| \times k.$
Where k<dim(h_i)
P_θ
with a dimension of
$|P_{idx}| \times \dim(h_i)$


Encoder-Decoder Model (e.g. BART)
PREFIX
x (source table)
z
Harry Potter , Education , Hogwarts
Activation
h_1 h_2 h_3 h_4 h_5 h_6 h_7 h_8
Indexing
1 2 3 4 5 6 7 8
$P_{idx} = [1, 2]$
$X_{idx} = [3, 4, 5, 6, 7, 8]$
PREFIX′
y (target utterance)
[SEP] Harry Potter is graduated from Hogwarts
h_9 h_{10} h_{11} h_{12} h_{13} h_{14} h_{15} h_{16} h_{17}
9 10 11 12 13 14 15 16 17
$P_{idx} \mathrel{+}= [9, 10]$
$Y_{idx} = [11, 12, 13, 14, 15, 16, 17]$

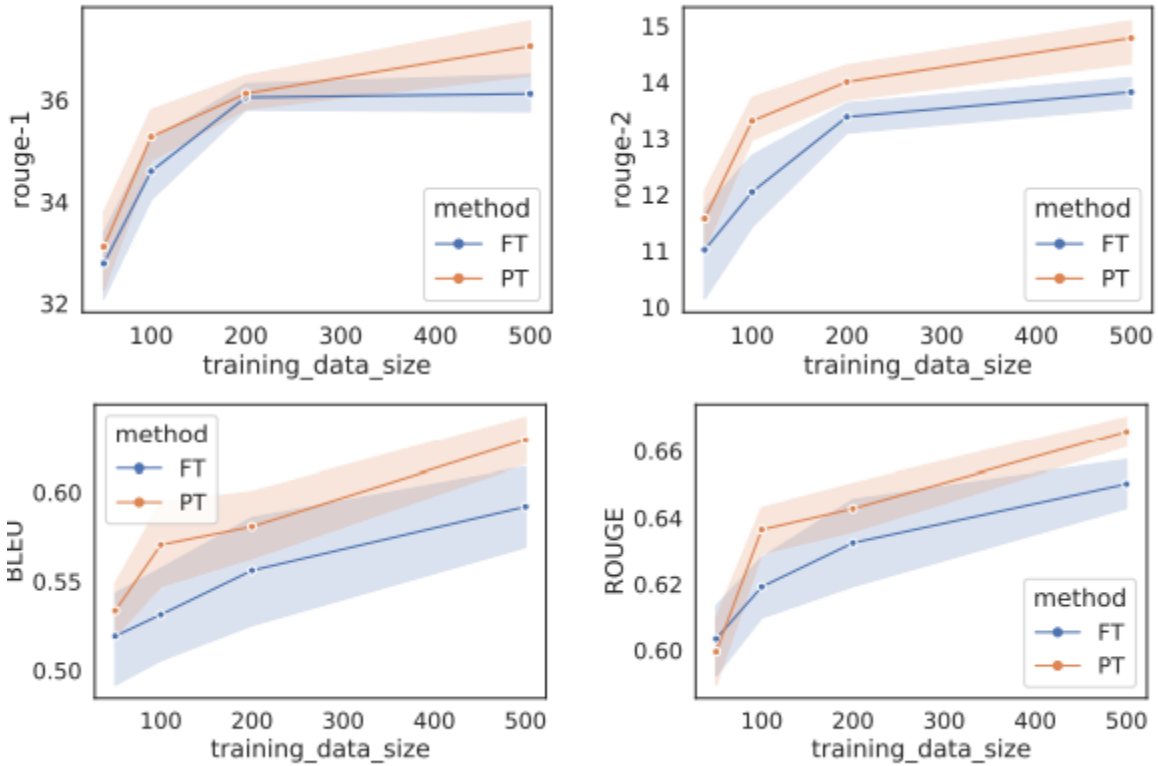
Ideal prefix length

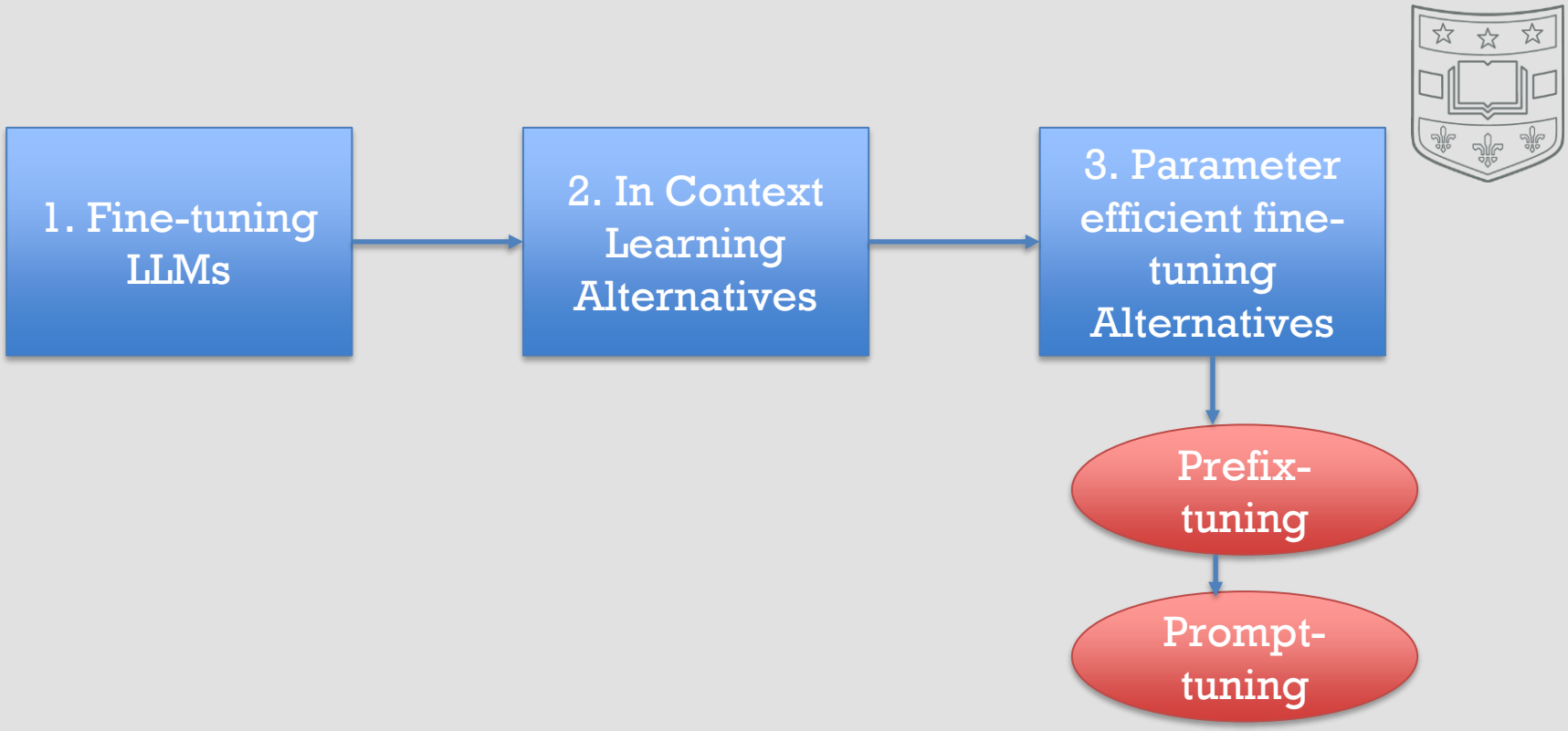


Performances

	E2E					WebNLG									DART					
	BLEU	NIST	MET	R-L	CIDEr	BLEU			MET			TER ↓			BLEU	MET	TER ↓	Mover	BERT	BLEURT
						S	U	A	S	U	A	S	U	A						
GPT-2$_{\text{MEDIUM}}$																				
FINE-TUNE	68.2	8.62	**46.2**	71.0	2.47	**64.2**	27.7	46.5	**0.45**	0.30	0.38	**0.33**	0.76	0.53	46.2	**0.39**	**0.46**	**0.50**	**0.94**	**0.39**
FT-TOP2	68.1	8.59	46.0	70.8	2.41	53.6	18.9	36.0	0.38	0.23	0.31	0.49	0.99	0.72	41.0	0.34	0.56	0.43	0.93	0.21
ADAPTER(3%)	68.9	8.71	46.1	71.3	2.47	60.4	**48.3**	54.9	0.43	**0.38**	**0.41**	0.35	**0.45**	**0.39**	45.2	0.38	**0.46**	**0.50**	**0.94**	**0.39**
ADAPTER(0.1%)	66.3	8.41	45.0	69.8	2.40	54.5	45.1	50.2	0.39	0.36	0.38	0.40	0.46	0.43	42.4	0.36	0.48	0.47	**0.94**	0.33
PREFIX(0.1%)	**69.7**	**8.81**	46.1	**71.4**	**2.49**	62.9	45.6	**55.1**	0.44	**0.38**	**0.41**	0.35	0.49	0.41	**46.4**	0.38	**0.46**	**0.50**	**0.94**	**0.39**
GPT-2$_{\text{LARGE}}$																				
FINE-TUNE	68.5	8.78	46.0	69.9	2.45	**65.3**	43.1	55.5	**0.46**	0.38	**0.42**	**0.33**	0.53	0.42	**47.0**	**0.39**	0.46	**0.51**	**0.94**	**0.40**
Prefix	**70.3**	**8.85**	**46.2**	**71.7**	**2.47**	63.4	**47.7**	**56.3**	0.45	**0.39**	**0.42**	0.34	**0.48**	**0.40**	46.7	**0.39**	**0.45**	**0.51**	**0.94**	**0.40**
SOTA	68.6	8.70	45.3	70.8	2.37	63.9	52.8	57.1	0.46	0.41	0.44	-	-	-	-	-	-	-	-	-

Performances under low data settings




1. Fine-tuning LLMs
2. In Context Learning Alternatives
3. Parameter efficient fine-tuning Alternatives
Prefix-tuning
Prompt-tuning

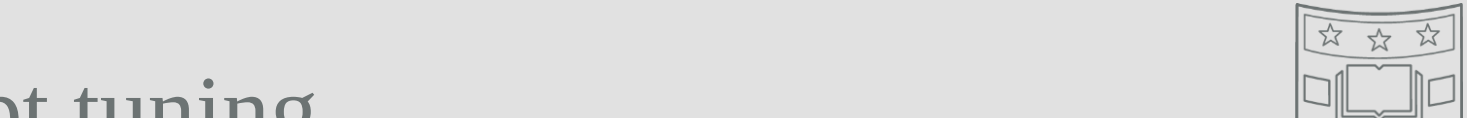
The Power of Scale for Parameter-Efficient Prompt Tuning

Brian Lester, Rami Al-Rfou, Noah Constant

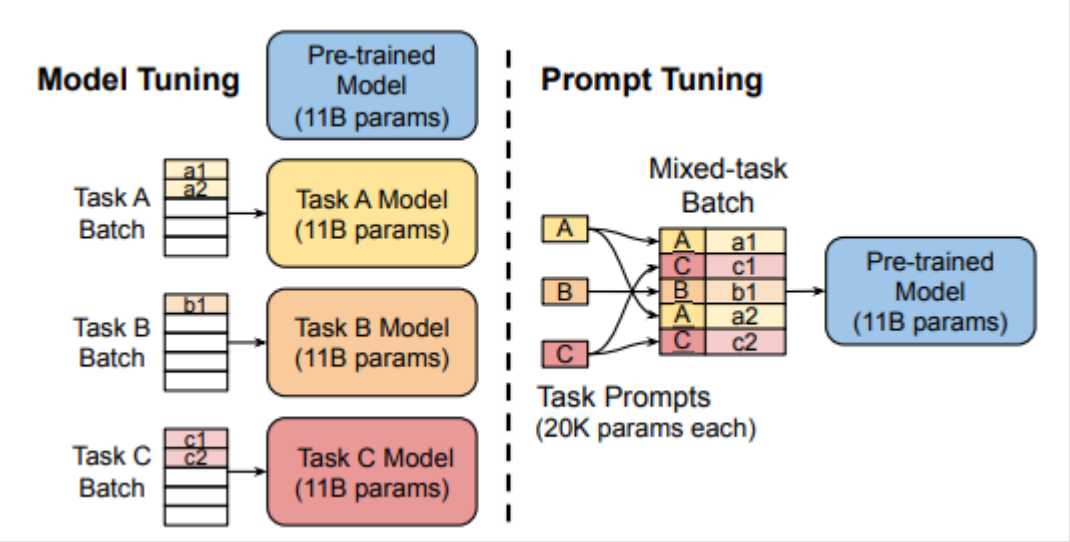
Abstract

In this work, we explore "prompt tuning," a simple yet effective mechanism for learning "soft prompts" to condition frozen language models to perform specific downstream tasks. Unlike the discrete text prompts used by GPT-3, soft prompts are learned through backpropagation and can be tuned to incorporate signals from any number of labeled examples. Our end-to-end learned approach outperforms GPT-3's few-shot learning by a large margin. More remarkably, through ablations on model size using T5, we show that prompt tuning becomes more competitive with scale: as models exceed billions of parameters, our method "closes the gap" and matches the strong performance of model tuning (where all model weights are tuned). This finding is especially relevant because large models are costly to share and serve and the ability to reuse one frozen model for multiple downstream tasks can ease this burden. Our method can be seen as a simplification of the recently proposed "prefix tuning" of Li and Liang (2021) and we provide a comparison to this and other similar approaches. Finally, we show that conditioning a frozen model with soft prompts confers benefits in robustness to domain transfer and enables efficient "prompt ensembling." We release code and model checkpoints to reproduce our experiments.

Prompt tuning



Same idea as Prefix-tuning!



Ways to initialize prompts

Random Uniform

Train from scratch, using random initialization

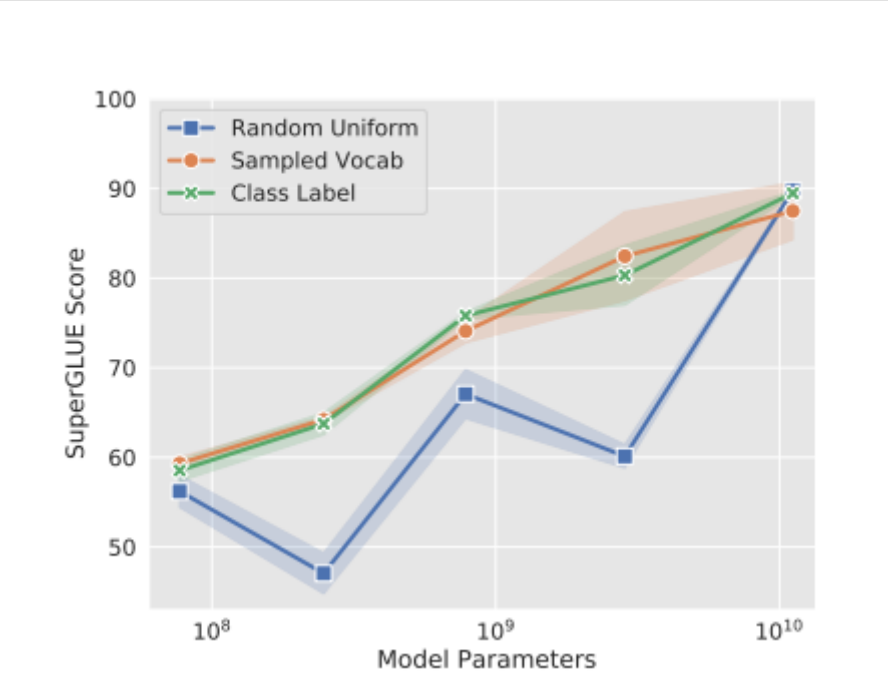
Sampled vocab

initialize each prompt token to an embedding drawn from the model's vocabulary.

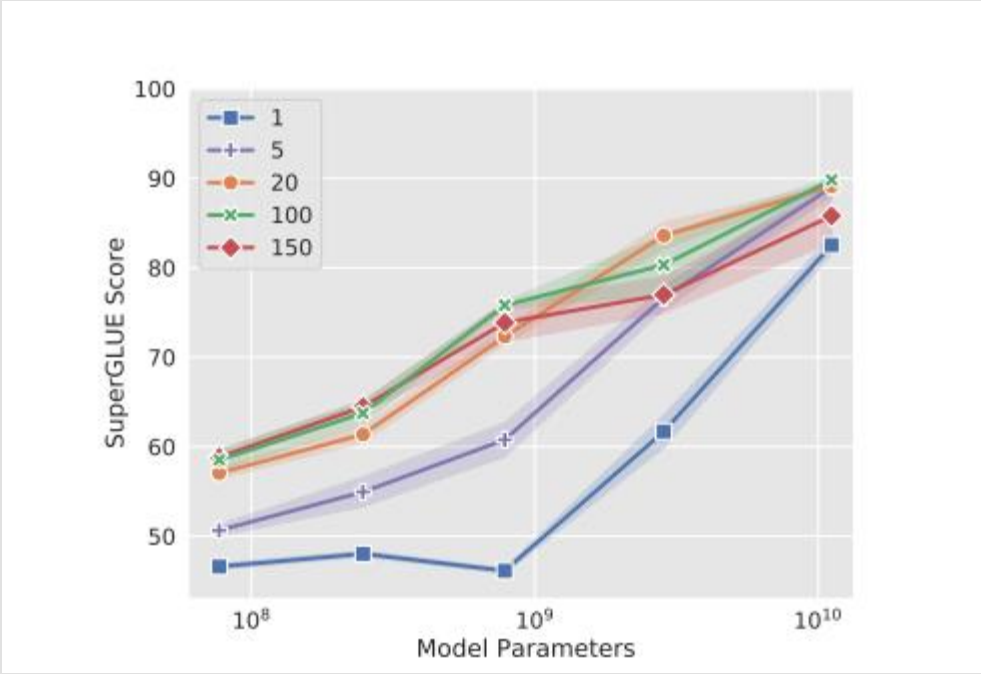
Class Label

initialize the prompt with embeddings that enumerate the output classes

Which one is the best strategy

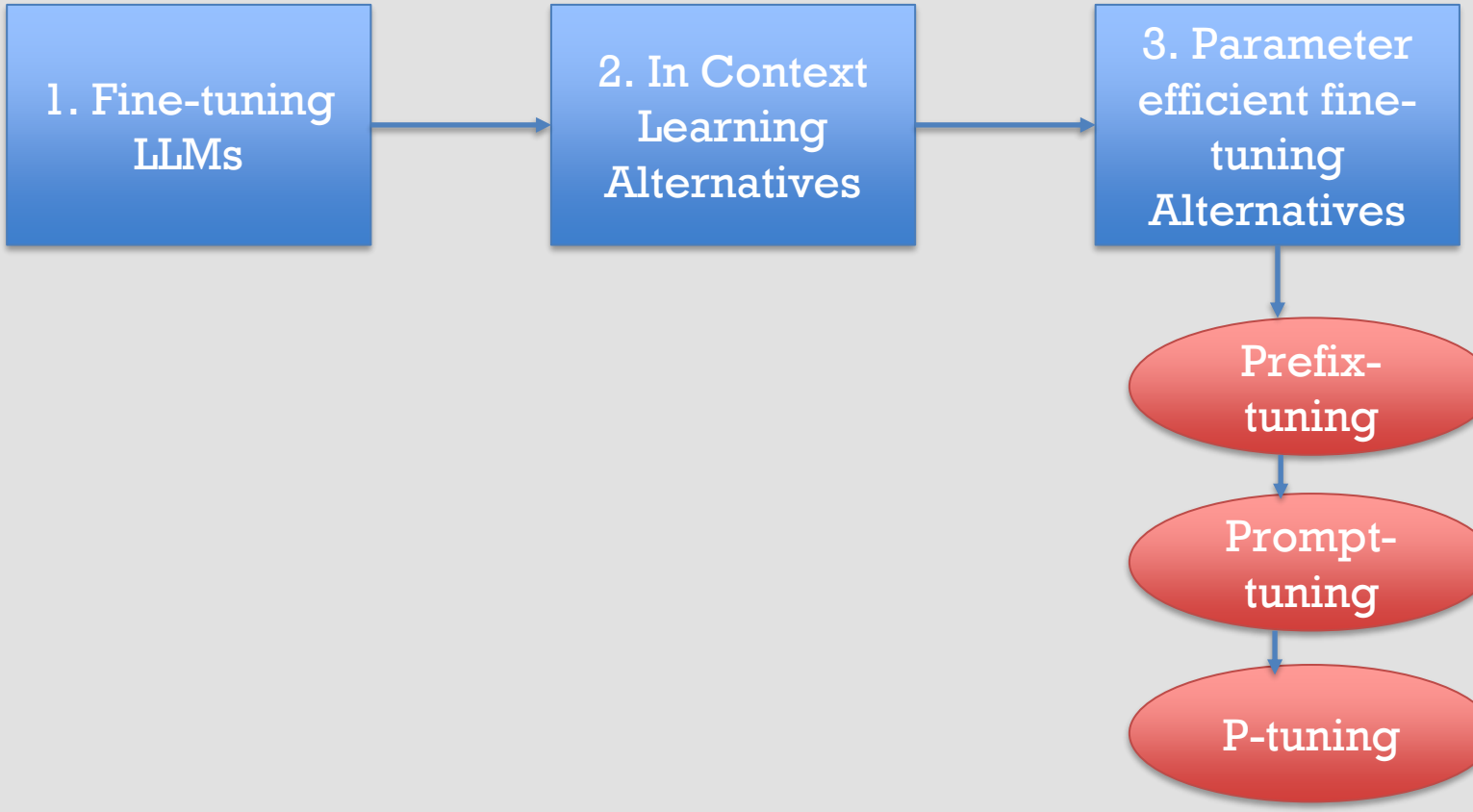


Ideal prompt length?



Performance

Dataset	Domain	Model	Prompt	Δ
SQuAD	Wiki	94.9 ±0.2	94.8 ±0.1	−0.1
TextbookQA	Book	54.3 ±3.7	**66.8** ±2.9	+12.5
BioASQ	Bio	77.9 ±0.4	**79.1** ±0.3	+1.2
RACE	Exam	59.8 ±0.6	**60.7** ±0.5	+0.9
RE	Wiki	88.4 ±0.1	**88.8** ±0.2	+0.4
DuoRC	Movie	**68.9** ±0.7	67.7 ±1.1	−1.2
DROP	Wiki	**68.9** ±1.7	67.1 ±1.9	−1.8


1. Fine-tuning LLMs
2. In Context Learning Alternatives
3. Parameter efficient fine-tuning Alternatives
Prefix-tuning
Prompt-tuning
P-tuning

KeAi
CHINESE ROOTS
GLOBAL IMPACT

AI Open

Available online 26 August 2023

In Press, Journal Pre-proof ? What's this?

GPT understands, too

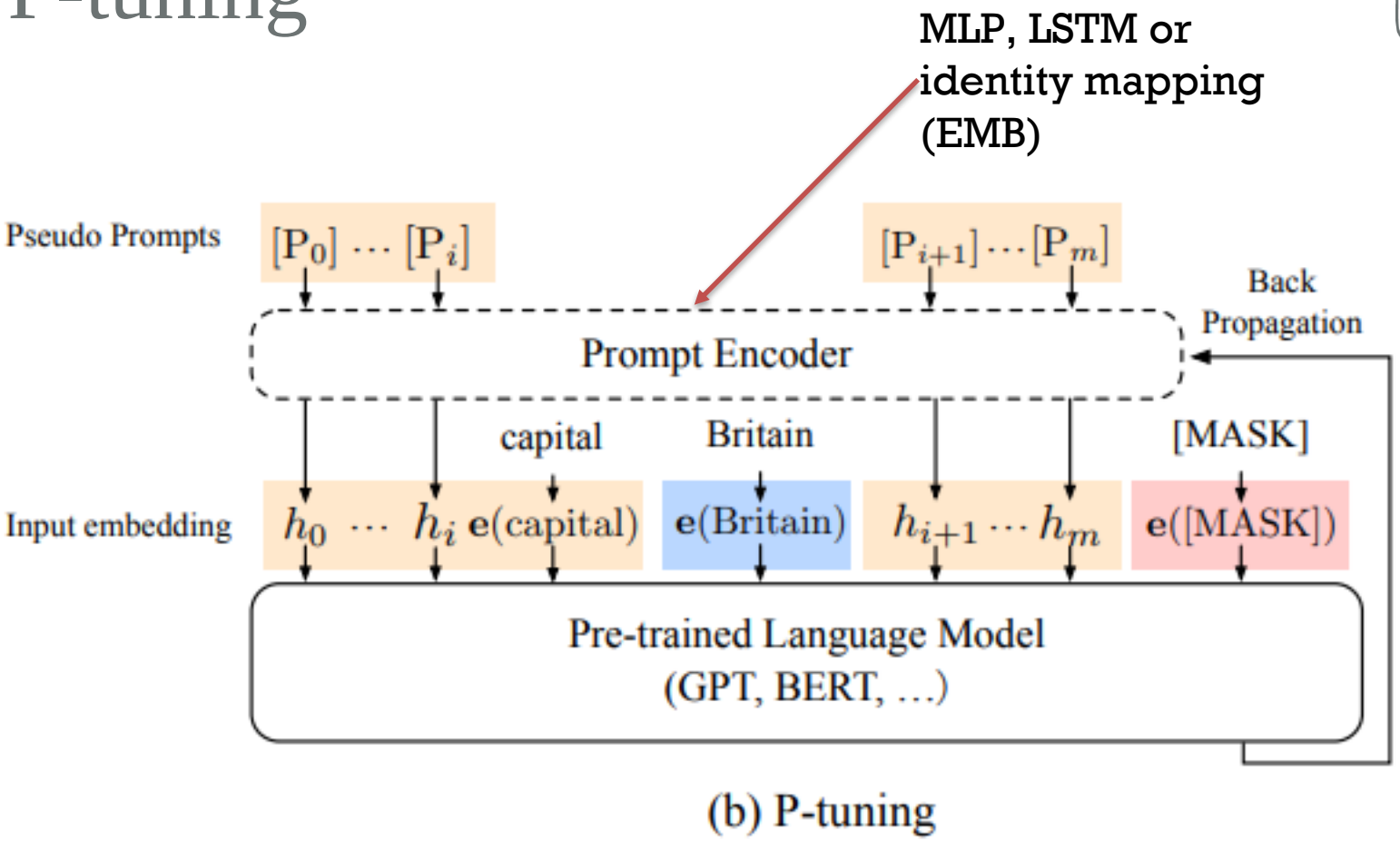
Xiao Liu [a 1], Yanan Zheng [a 1], Zhengxiao Du [a], Ming Ding [a], Yujie Qian [b], Zhilin Yang [a], Jie Tang [a]

Show more

+ Add to Mendeley Share Cite

https://doi.org/10.1016/j.aiopen.2023.08.012

P-tuning



(b) P-tuning

MLP, LSTM or Identity mapping (EMB)

Task	LSTM	MLP	EMB
WiC-ACC	56.27 ±1.54	55.25 ±3.09	53.96 ±3.23
CB-ACC.	81.70 ±7.49	88.39 ±3.72	82.59±3.69
CB-F1.	77.41 ±9.15	84.24±5.15	67.27±6.78
BoolQ-ACC.	75.41±3.09	76.46±2.84	76.87±1.69

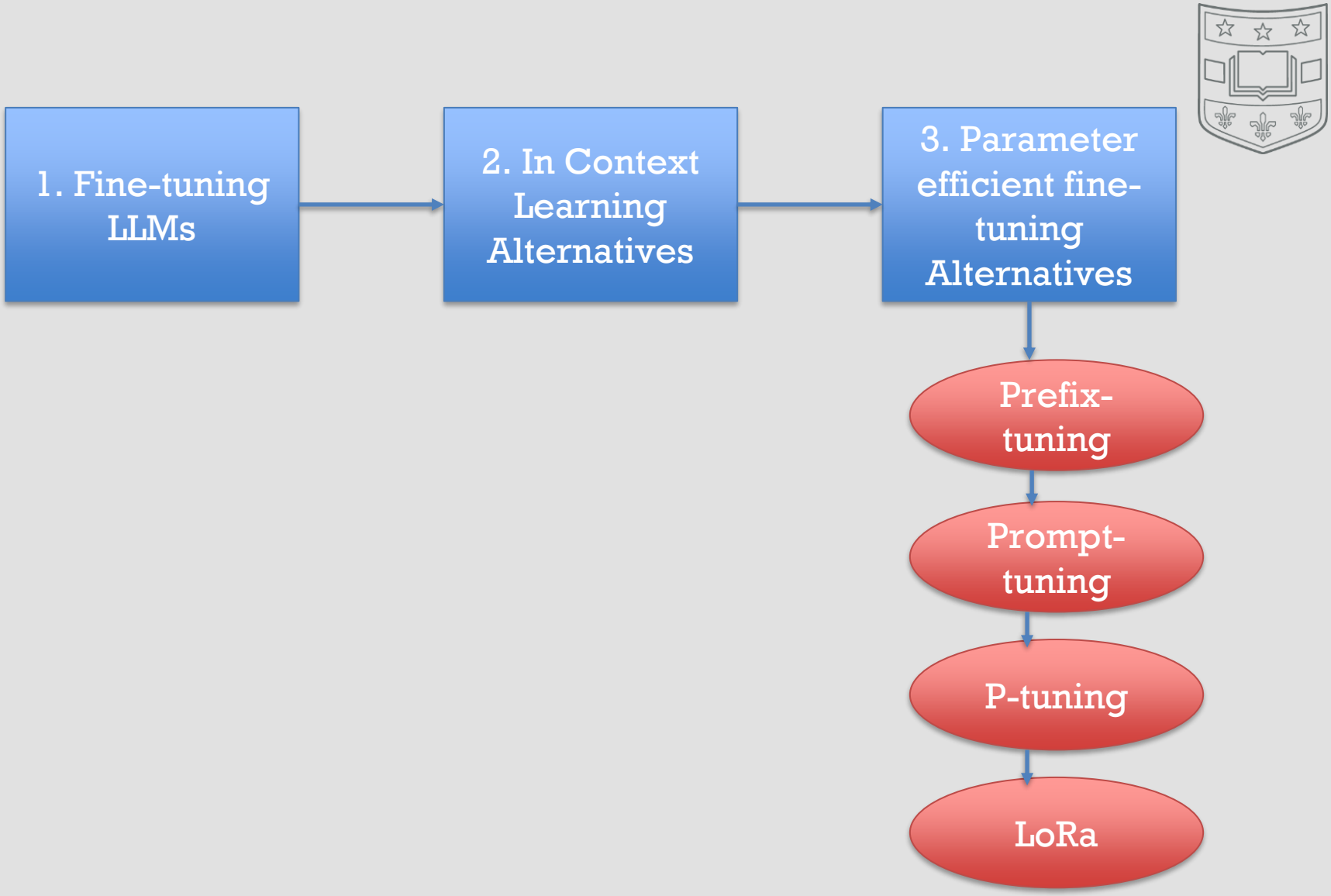
Performances

(a) Fully-supervised performance with base-scale models.

	Method	BoolQ (Acc.)	CB (Acc.)	CB (F1)	WiC (Acc.)	RTE (Acc.)	MultiRC (EM)	MultiRC (F1a)	WSC (Acc.)	COPA (Acc.)	Avg.
BERT-Base (109M)	CLS-FT	72.9	85.1	73.9	71.1	68.4	16.2	66.3	63.5	67.0	66.2
	PET-FT	73.7	87.5	90.8	67.9	70.4	13.7	62.5	60.6	70.0	67.1
	P-tuning	73.9	89.2	92.1	68.8	71.1	14.8	63.3	63.5	72.0	68.4
GPT2-Base (117M)	CLS-FT	71.2	78.6	55.8	65.5	67.8	17.4	65.8	63.0	64.4	63.0
	PET-FT	74.8	87.5	88.1	68.0	70.0	23.5	69.7	66.3	78.0	70.2
	P-tuning	75.0	91.1	93.2	68.3	70.8	23.5	69.8	63.5	76.0	70.4

(b) Fully-supervised performance with large-scale models.

	Method	BoolQ (Acc.)	CB (Acc.)	CB (F1)	WiC (Acc.)	RTE (Acc.)	MultiRC (EM)	MultiRC (F1a)	WSC (Acc.)	COPA (Acc.)	Avg.
BERT-Large (335M)	CLS-FT[1]	77.7	94.6	93.7	74.9	75.8	24.7	70.5	68.3	69.0	72.5
	PET-FT	77.2	91.1	93.5	70.5	73.6	17.7	67.0	80.8	75.0	73.1
	P-tuning	77.8	96.4	97.4	72.7	75.5	17.1	65.6	81.7	76.0	74.6
GPT2-Med. (345M)	CLS-FT	71.0	73.2	51.2	65.2	72.2	19.2	65.8	62.5	66.0	63.1
	PET-FT	78.3	96.4	97.4	70.4	72.6	32.1	74.4	73.0	80.0	74.9
	P-tuning	78.9	98.2	98.7	69.4	75.5	29.3	74.2	74.0	81.0	75.6


1. Fine-tuning LLMs
2. In Context Learning Alternatives
3. Parameter efficient fine-tuning Alternatives
Prefix-tuning
Prompt-tuning
P-tuning
LoRa

Computer Science > Computation and Language

[Submitted on 17 Jun 2021 (v1), last revised 16 Oct 2021 (this version, v2)]

LoRA: Low-Rank Adaptation of Large Language Models

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen

An important paradigm of natural language processing consists of large-scale pre-training on general domain data and adaptation
feasible. Using GPT-3 175B as an example -- deploying independent instances of fine-tuned models, each with 175B parameters,
trainable rank decomposition matrices into each layer of the Transformer architecture, greatly reducing the number of trainable par
by 10,000 times and the GPU memory requirement by 3 times. LoRA performs on-par or better than fine-tuning in model quality on
adapters, no additional inference latency. We also provide an empirical investigation into rank-deficiency in language model adapta
provide our implementations and model checkpoints for RoBERTa, DeBERTa, and GPT-2 at this https URL.

Submitted on 22 Dec 2020]

Intrinsic Dimensionality Explains the Effectiveness of Language Model Fine-Tuning

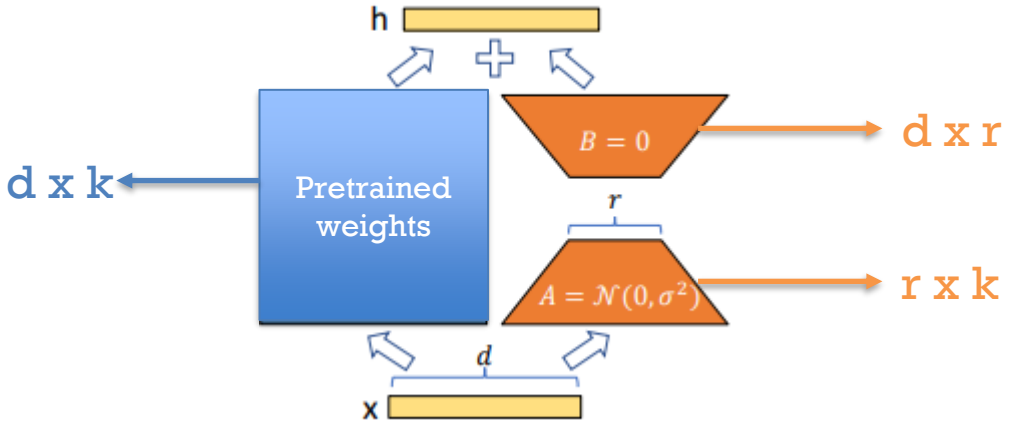
Armen Aghajanyan, Luke Zettlemoyer, Sonal Gupta

Although pretrained language models can be fine-tuned to produce state-of-the-art results for a very wide range of language understanding tasks, the dynamics of this process are not well understood, especially in the low data regime. Why can we use relatively vanilla gradient descent algorithms (e.g., without strong regularization) to tune a model with hundreds of millions of parameters on datasets with only hundreds or thousands of labeled examples? In this paper, we argue that analyzing fine-tuning through the lens of intrinsic dimension provides us with empirical and theoretical intuitions to explain this remarkable phenomenon. We empirically show that common pre-trained models have a very low intrinsic dimension; in other words, there exists a low dimension reparameterization that is as effective for fine-tuning as the full parameter space. For example, by optimizing only 200 trainable parameters randomly projected back into the full space, we can tune a RoBERTa model to achieve 90\% of the full parameter performance levels on MRPC. Furthermore, we empirically show that pre-training implicitly minimizes intrinsic dimension and, perhaps surprisingly, larger models tend to have lower intrinsic dimension after a fixed number of pre-training updates, at least in part explaining their extreme effectiveness. Lastly, we connect intrinsic dimensionality with low dimensional task representations and compression based generalization bounds to provide intrinsic-dimension-based generalization bounds that are independent of the full parameter count.


Low “intrinsic-dimension”
Low intrinsic rank?

Low "intrinsic-dimension"

Low intrinsic rank?



where $r \ll \min(d, k)$.

$$\Delta W = \alpha AB$$

Scaling factor

$$h = W_0 x + \Delta W x = W_0 x + \alpha ABx$$

Model & Method	# Trainable Parameters	MNLI	SST-2	MRPC	CoLA	QNLI	QQP	RTE	STS-B	Avg.
RoB_{base} (FT)*	125.0M	**87.6**	94.8	90.2	**63.6**	92.8	**91.9**	78.7	91.2	86.4
RoB_{base} (BitFit)*	0.1M	84.7	93.7	**92.7**	62.0	91.8	84.0	81.5	90.8	85.2
RoB_{base} (Adpt^{D})*	0.3M	$87.1_{\pm.0}$	$94.2_{\pm.1}$	$88.5_{\pm1.1}$	$60.8_{\pm.4}$	$93.1_{\pm.1}$	$90.2_{\pm.0}$	$71.5_{\pm2.7}$	$89.7_{\pm.3}$	84.4
RoB_{base} (Adpt^{D})*	0.9M	$87.3_{\pm.1}$	$94.7_{\pm.3}$	$88.4_{\pm.1}$	$62.6_{\pm.9}$	$93.0_{\pm.2}$	$90.6_{\pm.0}$	$75.9_{\pm2.2}$	$90.3_{\pm.1}$	85.4
RoB_{base} (LoRA)	0.3M	$87.5_{\pm.3}$	$\mathbf{95.1}_{\pm.2}$	$89.7_{\pm.7}$	$63.4_{\pm1.2}$	$\mathbf{93.3}_{\pm.3}$	$90.8_{\pm.1}$	$\mathbf{86.6}_{\pm.7}$	$\mathbf{91.5}_{\pm.2}$	**87.2**
$\text{RoB}_{\text{large}}$ (FT)*	355.0M	90.2	**96.4**	**90.9**	68.0	94.7	**92.2**	86.6	92.4	88.9
$\text{RoB}_{\text{large}}$ (LoRA)	0.8M	$\mathbf{90.6}_{\pm.2}$	$96.2_{\pm.5}$	$\mathbf{90.9}_{\pm1.2}$	$\mathbf{68.2}_{\pm1.9}$	$\mathbf{94.9}_{\pm.3}$	$91.6_{\pm.1}$	$\mathbf{87.4}_{\pm2.5}$	$\mathbf{92.6}_{\pm.2}$	**89.0**
$\text{RoB}_{\text{large}}$ (Adpt^{P})†	3.0M	$90.2_{\pm.3}$	$96.1_{\pm.3}$	$90.2_{\pm.7}$	$\mathbf{68.3}_{\pm1.0}$	$\mathbf{94.8}_{\pm.2}$	$\mathbf{91.9}_{\pm.1}$	$83.8_{\pm2.9}$	$92.1_{\pm.7}$	88.4
$\text{RoB}_{\text{large}}$ (Adpt^{P})†	0.8M	$\mathbf{90.5}_{\pm.3}$	$\mathbf{96.6}_{\pm.2}$	$89.7_{\pm1.2}$	$67.8_{\pm2.5}$	$\mathbf{94.8}_{\pm.3}$	$91.7_{\pm.2}$	$80.1_{\pm2.9}$	$91.9_{\pm.4}$	87.9
$\text{RoB}_{\text{large}}$ (Adpt^{H})†	6.0M	$89.9_{\pm.5}$	$96.2_{\pm.3}$	$88.7_{\pm2.9}$	$66.5_{\pm4.4}$	$94.7_{\pm.2}$	$92.1_{\pm.1}$	$83.4_{\pm1.1}$	$91.0_{\pm1.7}$	87.8
$\text{RoB}_{\text{large}}$ (Adpt^{H})†	0.8M	$90.3_{\pm.3}$	$96.3_{\pm.5}$	$87.7_{\pm1.7}$	$66.3_{\pm2.0}$	$94.7_{\pm.2}$	$91.5_{\pm.1}$	$72.9_{\pm2.9}$	$91.5_{\pm.5}$	86.4
$\text{RoB}_{\text{large}}$ (LoRA)†	0.8M	$\mathbf{90.6}_{\pm.2}$	$96.2_{\pm.5}$	$\mathbf{90.2}_{\pm1.0}$	$68.2_{\pm1.9}$	$\mathbf{94.8}_{\pm.3}$	$91.6_{\pm.2}$	$\mathbf{85.2}_{\pm1.1}$	$\mathbf{92.3}_{\pm.5}$	**88.6**
DeB_{XXL} (FT)*	1500.0M	91.8	**97.2**	92.0	72.0	**96.0**	92.7	93.9	92.9	91.1
DeB_{XXL} (LoRA)	4.7M	$\mathbf{91.9}_{\pm.2}$	$96.9_{\pm.2}$	$\mathbf{92.6}_{\pm.6}$	$\mathbf{72.4}_{\pm1.1}$	$\mathbf{96.0}_{\pm.1}$	$\mathbf{92.9}_{\pm.1}$	$\mathbf{94.9}_{\pm.4}$	$\mathbf{93.0}_{\pm.2}$	**91.3**

What is the right *r*?
Which weights should LoRa apply to?

	Weight Type	$r = 1$	$r = 2$	$r = 4$	$r = 8$	$r = 64$
WikiSQL(±0.5%)	W_q	68.8	69.6	70.5	70.4	70.0
	W_q, W_v	73.4	73.3	73.7	73.8	73.5
	W_q, W_k, W_v, W_o	74.1	73.7	74.0	74.0	73.9
MultiNLI (±0.1%)	W_q	90.7	90.9	91.1	90.7	90.7
	W_q, W_v	91.3	91.4	91.3	91.6	91.4
	W_q, W_k, W_v, W_o	91.2	91.7	91.7	91.5	91.4

Idea: with a small *r* even just on W_q and W_v , you can achieve great performances

How much more computational resources conserved?

- For GPT3-175B
  - VRAM consumption: 1.2TB --> 350GB
  - 25% speed up in time

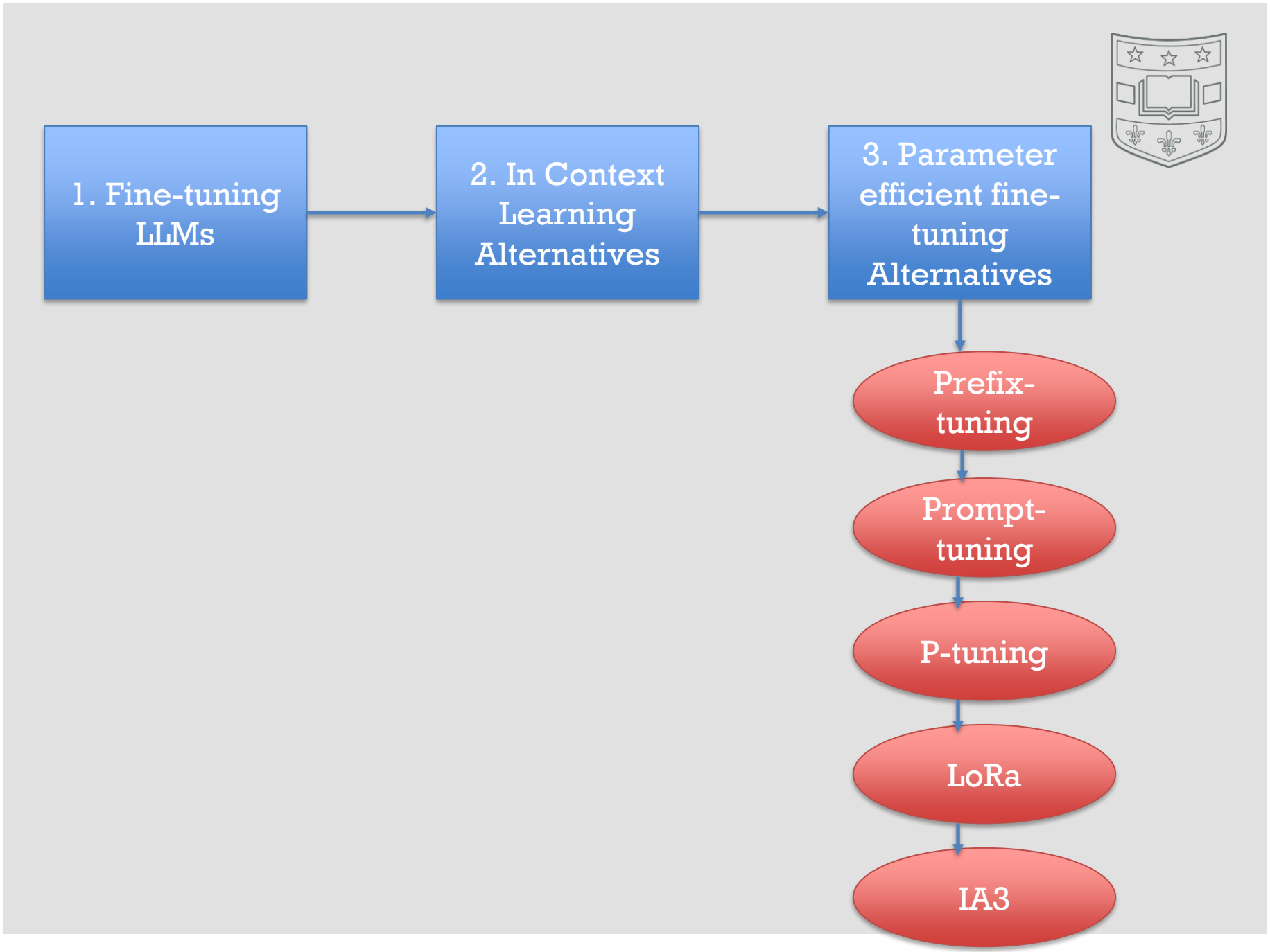
Few-Shot Parameter-Efficient Fine-Tuning is Better and Cheaper than In-Context Learning

Part of Advances in Neural Information Processing Systems 35 (NeurIPS 2022) Main Conference Track

Bibtex Paper Supplemental

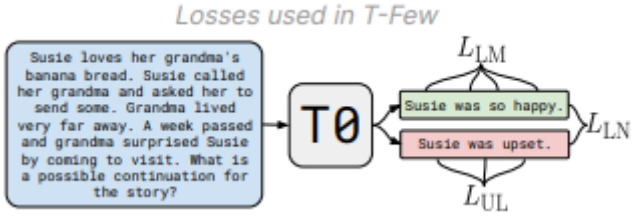
Authors

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, Colin A. Raffel


1. Fine-tuning LLMs
2. In Context Learning Alternatives
3. Parameter efficient fine-tuning Alternatives
Prefix-tuning
Prompt-tuning
P-tuning
LoRa
IA3

Introduce 2 losses

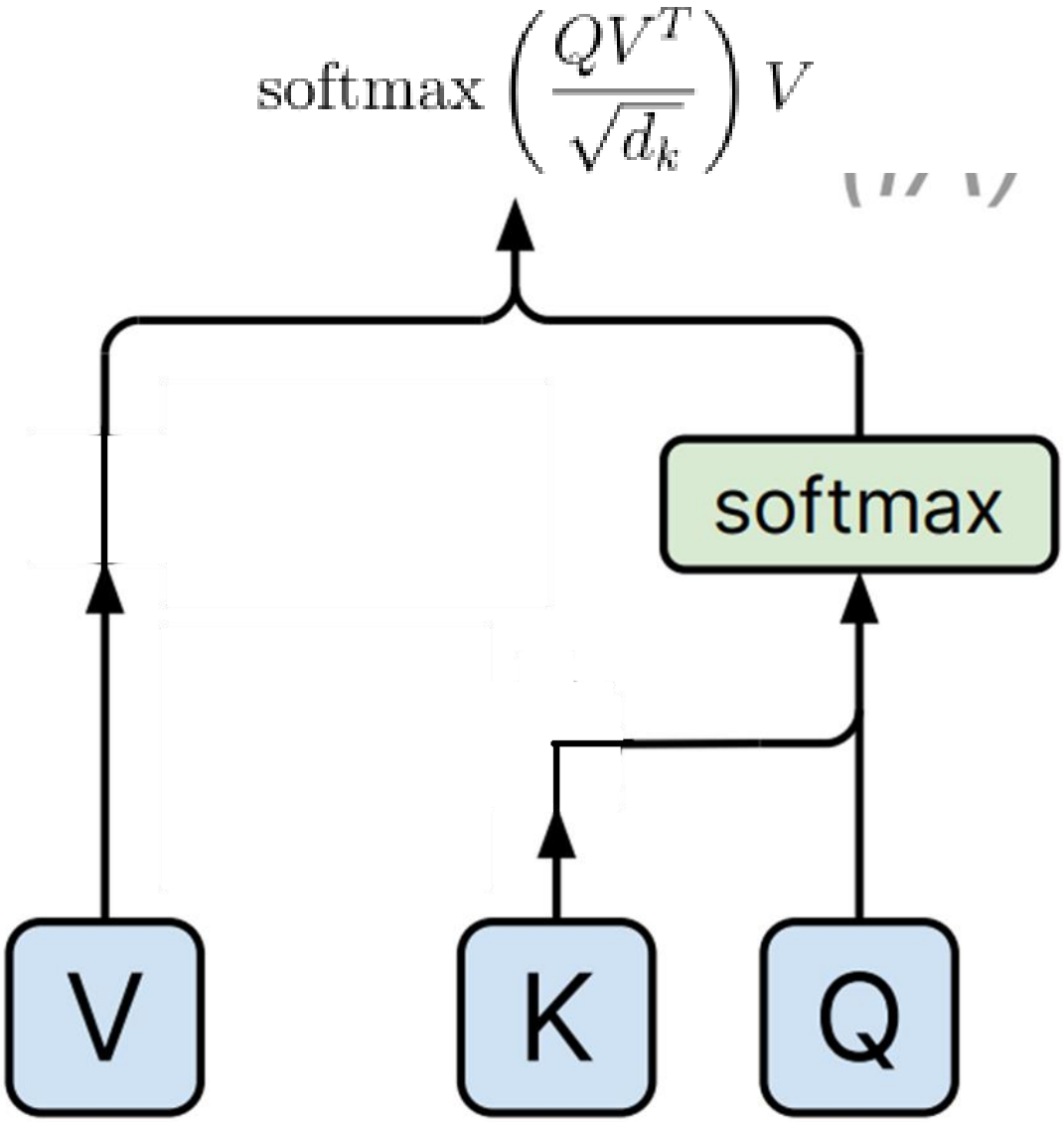
- Unlikelihood loss $$L_{\mathrm{UL}} = -\frac{\sum_{n=1}^{N}\sum_{t=1}^{T^{(n)}} \log(1 - p(\hat{y}_i^{(n)}|\mathbf{x}, \hat{y}_{<t}^{(n)}))}{\sum_{n=1}^{N} T^{(n)}}$$
- Length normalized Loss^ $$L_{\mathrm{LN}} = -\log \frac{\exp(\beta(\mathbf{x}, \mathbf{y}))}{\exp(\beta(\mathbf{x}, \mathbf{y})) + \sum_{n=1}^{N} \exp(\beta(\mathbf{x}, \hat{\mathbf{y}}^{(n)}))}$$



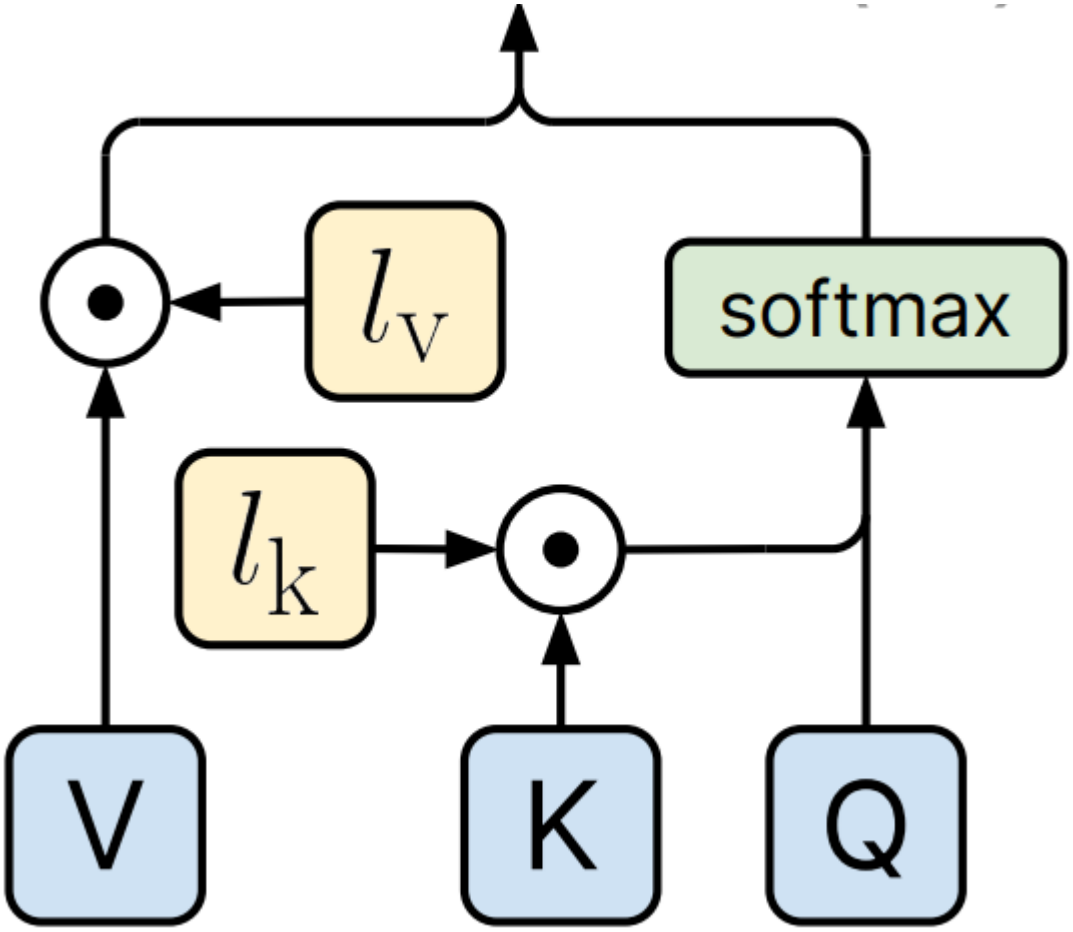
^ basically the log-probability relative to other tokens

Introduce IA3 training

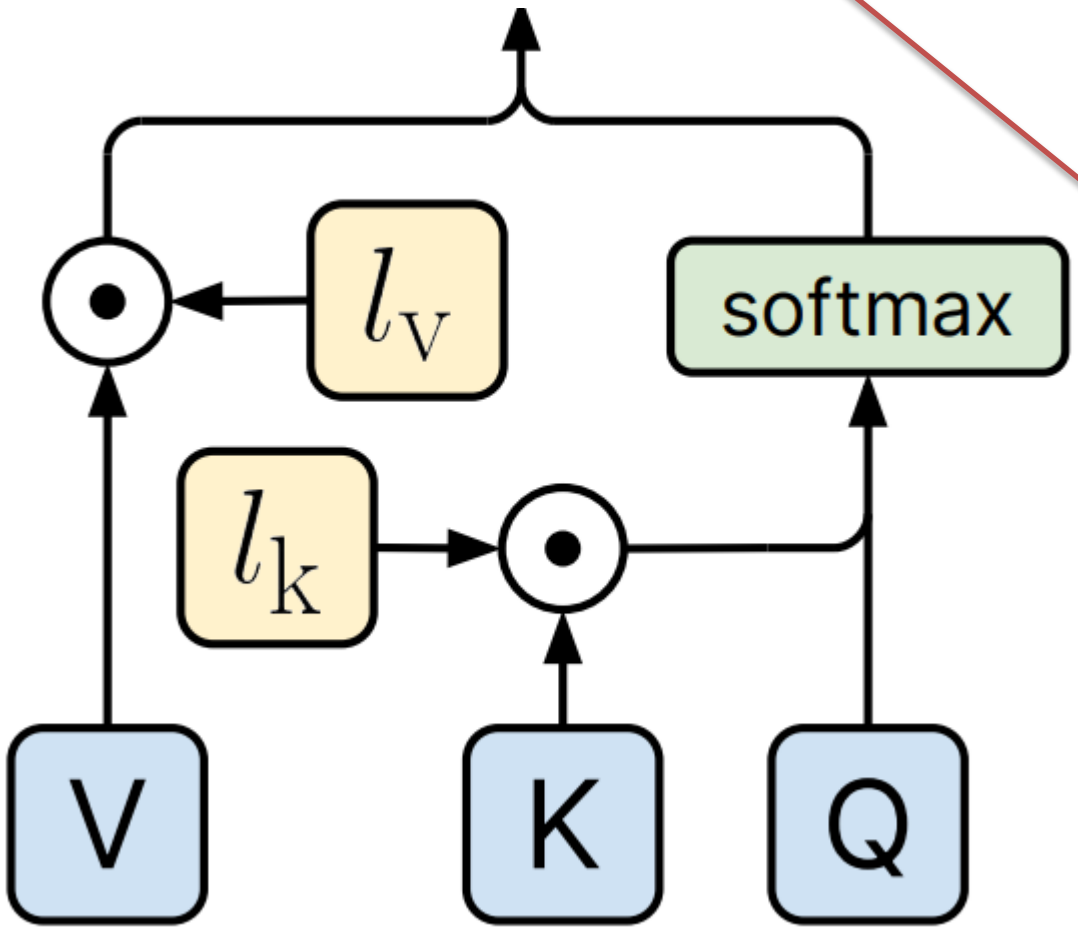
- Short for "Infused Adapter by Inhibiting and Amplifying Inner Activations"
- Introduce 3 learned vectors


$\text{softmax}\left(\frac{QV^T}{\sqrt{d_k}}\right)V$
softmax
V
K
Q

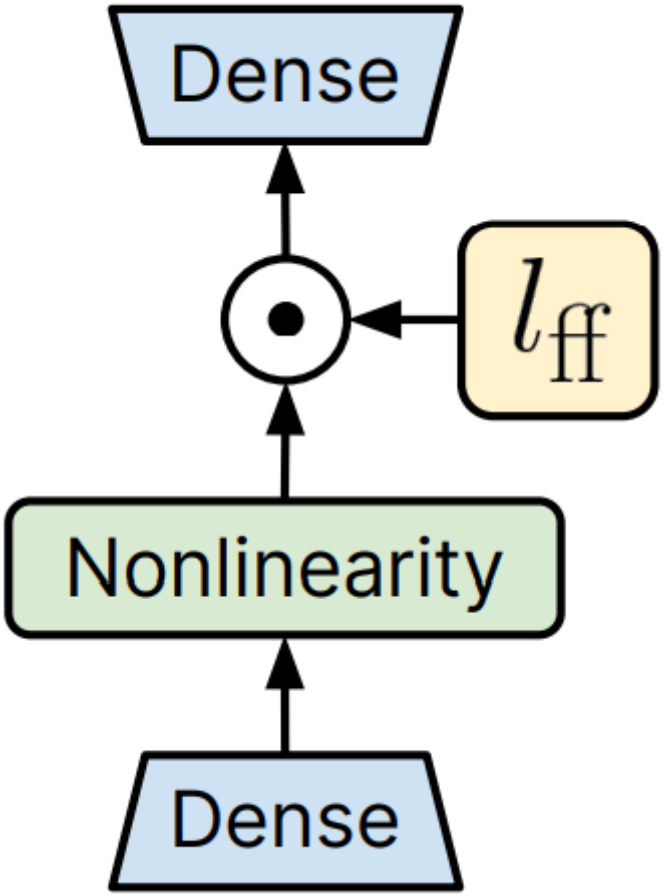
$$\text{softmax}\left(\frac{Q(l_{\text{k}} \odot K^T)}{\sqrt{d_k}}\right)(l_{\text{v}} \odot V)$$



$$\text{softmax}\left(\frac{Q(l_{\text{k}} \odot K^T)}{\sqrt{d_k}}\right)(l_{\text{v}} \odot V)$$

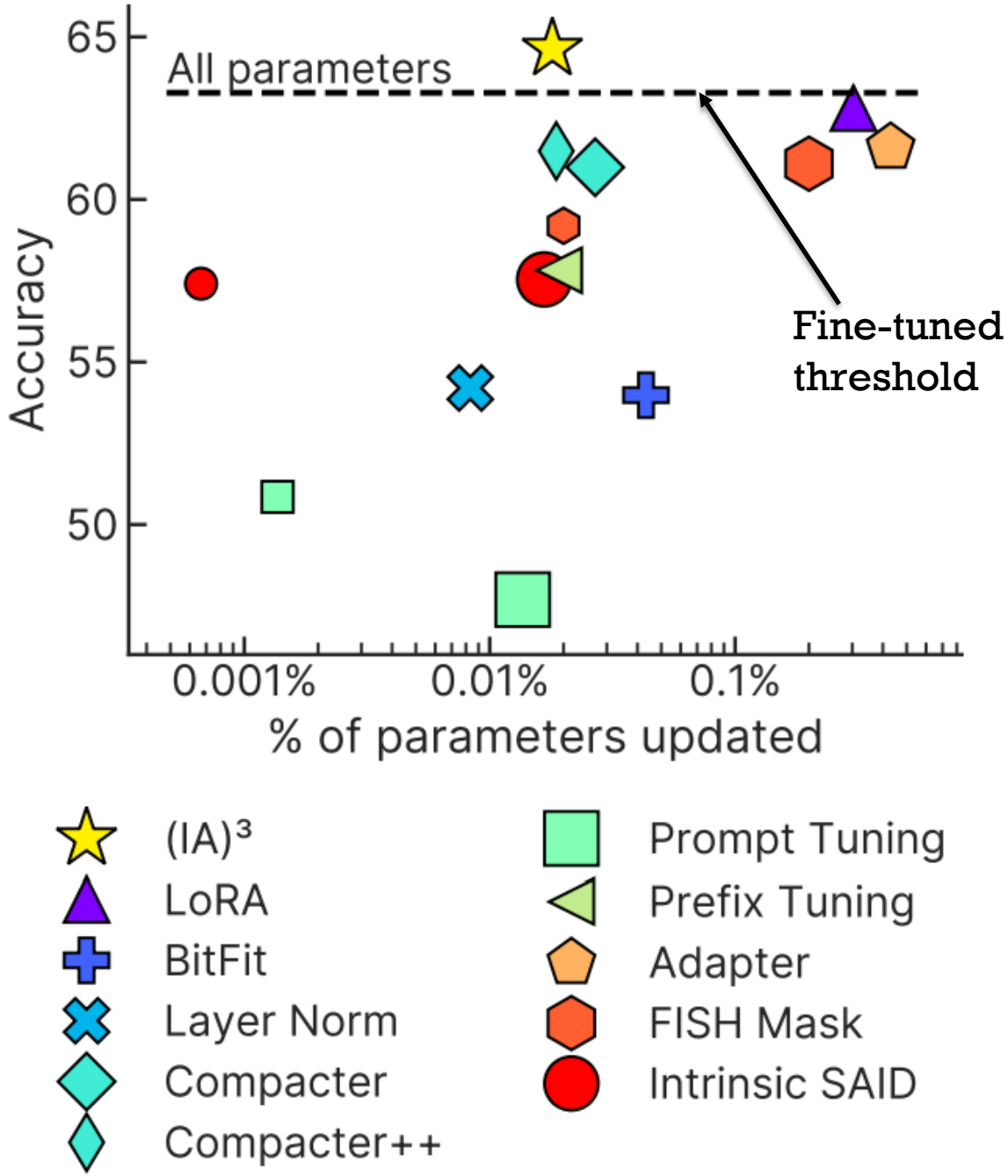


Learned task-specific vector initialized to 1's


Dense
l_{ff}
Nonlinearity
Dense

Number of parameters required decreases

From the encoder	From the decoder
$L(d_k + d_v + d_{\text{ff}})$	$L(2d_k + 2d_v + d_{\text{ff}})$


65
60
55
50
Accuracy
All parameters
Fine-tuned threshold
0.001%
0.01%
0.1%
% of parameters updated
(IA)³
LoRA
BitFit
Layer Norm
Compacter
Compacter++
Prompt Tuning
Prefix Tuning
Adapter
FISH Mask
Intrinsic SAID

Performs better than humans as well with few-shot learning!

Method	Acc.
T-Few	75.8%
Human baseline [2]	73.5%
PET [50]	69.6%
SetFit [51]	66.9%
GPT-3 [4]	62.7%

Table 2: Top-5 best methods on RAFT as of writing. T-Few is the first method to outperform the human baseline and achieves over 6% higher accuracy than the next-best method.

Cheaper as well for prompting