

# **LLMs as few-shot learners**

**Advanced NLP: Summer 2023**

**Anoop Sarkar**

**"Language provides a natural domain for the study of artificial intelligence, as the vast majority of reasoning tasks can be efficiently expressed and evaluated in language, and the world's text provides a wealth of data for unsupervised learning via generative modeling."**

**- OpenAI**

---

# Improving Language Understanding by Generative Pre-Training

---

GPT1

**Alec Radford**  
OpenAI  
alec@openai.com

**Karthik Narasimhan**  
OpenAI  
karthikn@openai.com

**Tim Salimans**  
OpenAI  
tim@openai.com

**Ilya Sutskever**  
OpenAI  
ilyasu@openai.com

# GPT1

## Pre-training an autoregressive language model

BooksCorpus: 7K  
unpublished books  
(1B words)

- Start with a large amount of unlabeled data  $\mathcal{U} = \{u_1, \dots, u_n\}$

- Pre-training objective: Maximize the likelihood of predicting the next token

$$L_i(\mathcal{U}) = \sum_i \log P(u_i | u_{i-k}, \dots, u_{i-1}; \Theta)$$

$U = (u_{-k}, \dots, u_{-1})$  is the context vector of tokens

- This is equivalent to training a Transformer decoder

$n$  is the number of Transformer layers

$$h_0 = U \boxed{W_e} + W_p$$

$W_e$  is the token embedding matrix

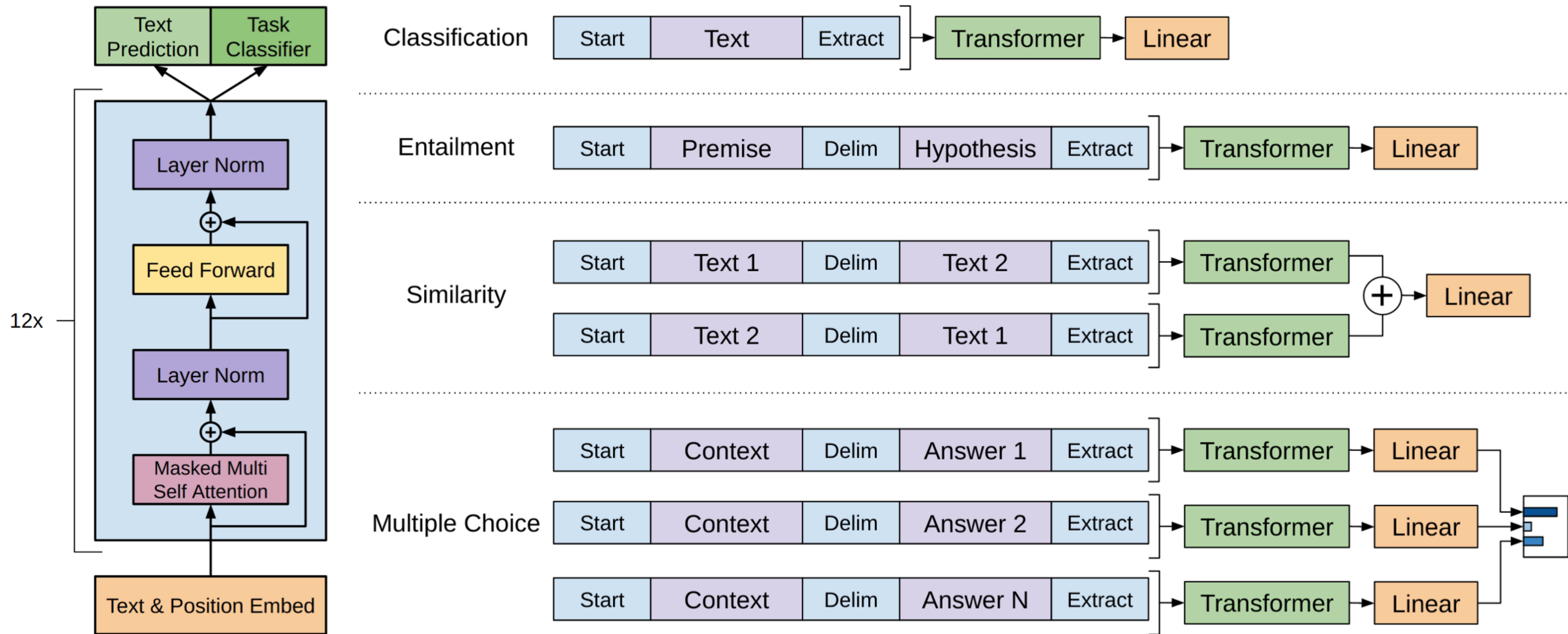
$$h_\ell = \text{transformer\_block}(h_{\ell-1}) \forall \ell \in [1, n]$$

$W_p$  is the position embedding matrix

$$P(u) = \text{softmax}(h_n \boxed{W_e^T})$$

- Directionality is needed to generate a well-formed probability distribution





This setup was for fine-tuning GPT1 but also works for in-context learning in GPT2 and GPT3.

The GPT2 paper

---

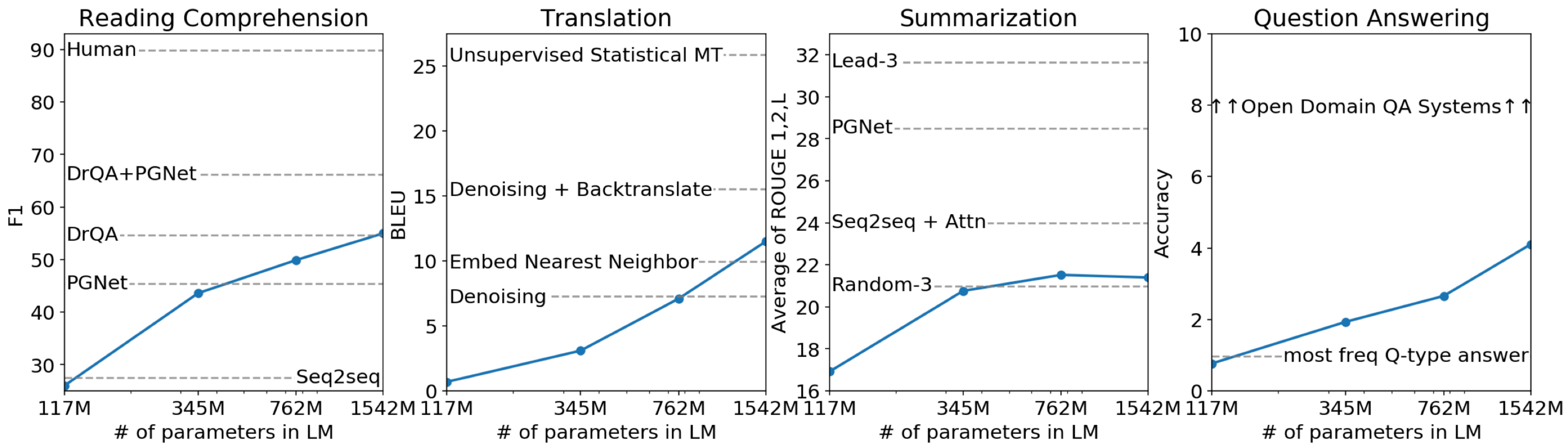
# Language Models are Unsupervised Multitask Learners

---

**Alec Radford** \*<sup>1</sup> **Jeffrey Wu** \*<sup>1</sup> **Rewon Child**<sup>1</sup> **David Luan**<sup>1</sup> **Dario Amodei** \*\*<sup>1</sup> **Ilya Sutskever** \*\*<sup>1</sup>

[https://cdn.openai.com/better-language-models/  
language\\_models\\_are\\_unsupervised\\_multitask\\_learners.pdf](https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf)

Feb 2019



# WebText corpus

- Train on web scale corpus but with more reliable data compared to the CommonCrawl.
- English-only, so language detection is used
- Outgoing links from reddit (with at least 3 karma)
- No reddit data was used, instead use the content of the web sites linked on reddit discussions
- 8M documents with 40GB of text

Language detection: <https://github.com/CLD2Owners/cld2>

News site scraping: <https://github.com/codelucas/newspaper>

---

”I’m not the cleverest man in the world, but like they say in French: **Je ne suis pas un imbecile** [I’m not a fool].

In a now-deleted post from Aug. 16, Soheil Eid, Tory candidate in the riding of Joliette, wrote in French: ”**Mentez mentez, il en restera toujours quelque chose,**” which translates as, ”**Lie lie and something will always remain.**”

“I hate the word ‘**perfume,**’” Burr says. ‘It’s somewhat better in French: ‘**parfum.**’

If listened carefully at 29:55, a conversation can be heard between two guys in French: “-**Comment on fait pour aller de l’autre coté? -Quel autre coté?**”, which means “- **How do you get to the other side? - What side?**”.

If this sounds like a bit of a stretch, consider this question in French: **As-tu aller au cinéma?**, or **Did you go to the movies?**, which literally translates as Have-you to go to movies/theater?

“**Brevet Sans Garantie Du Gouvernement**”, translated to English: “**Patented without government warranty**”.

---

*Table 1.* Examples of naturally occurring demonstrations of English to French and French to English translation found throughout the WebText training set.

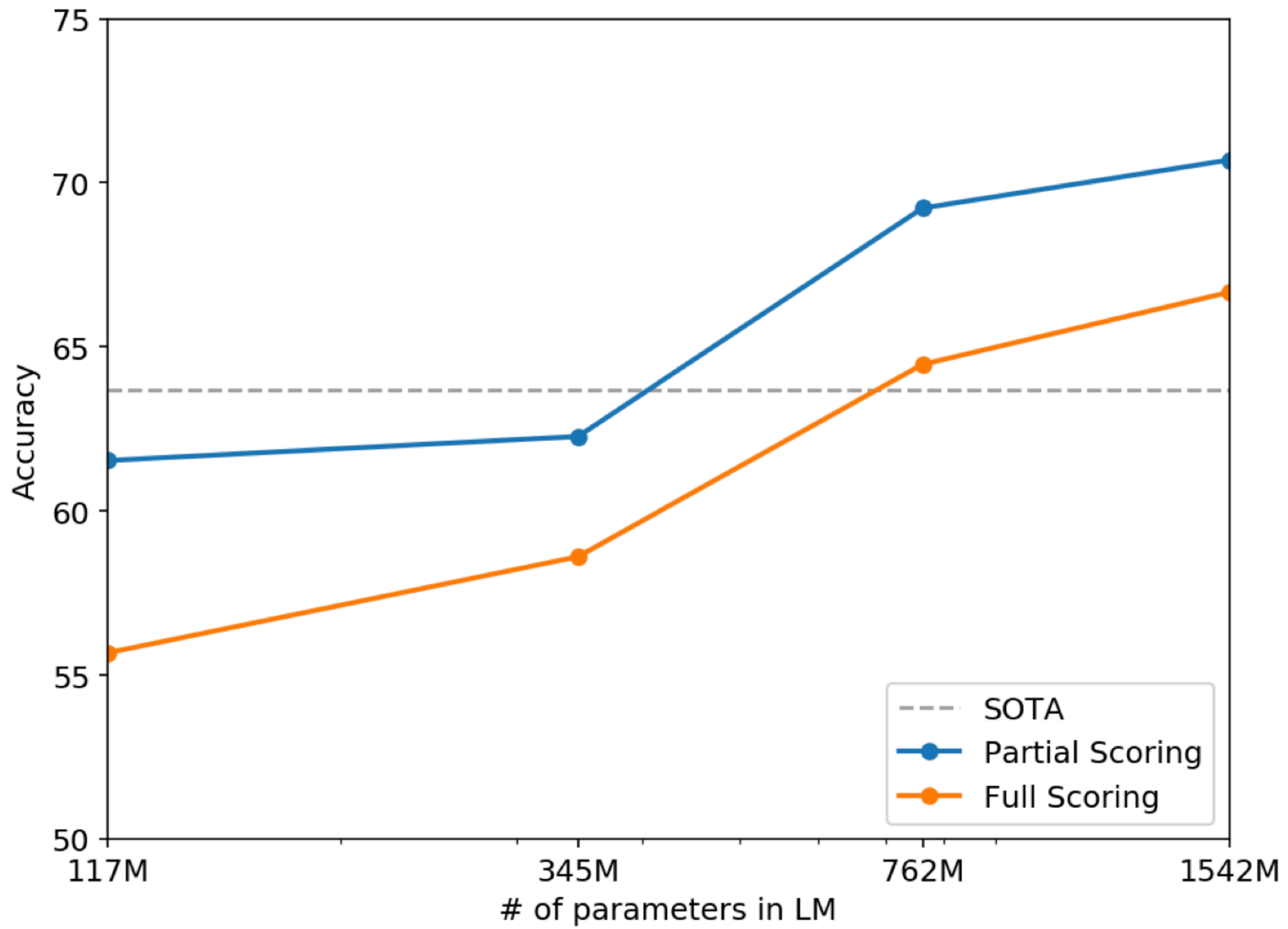
Parameters	Layers	$d_{model}$
117M	12	768
345M	24	1024
762M	36	1280
1542M	48	1600

*Table 2.* Architecture hyperparameters for the 4 model sizes.



## Perplexity Results

	LAMBADA (PPL)	LAMBADA (ACC)	CBT-CN (ACC)	CBT-NE (ACC)	WikiText2 (PPL)	PTB (PPL)	enwik8 (BPB)	text8 (BPC)	WikiText103 (PPL)	1BW (PPL)
SOTA	99.8	59.23	85.7	82.3	39.14	46.54	0.99	1.08	18.3	<b>21.8</b>
117M	<b>35.13</b>	45.99	<b>87.65</b>	<b>83.4</b>	<b>29.41</b>	65.85	1.16	1.17	37.50	75.20
345M	<b>15.60</b>	55.48	<b>92.35</b>	<b>87.1</b>	<b>22.76</b>	47.33	1.01	<b>1.06</b>	26.37	55.72
762M	<b>10.87</b>	<b>60.12</b>	<b>93.45</b>	<b>88.0</b>	<b>19.93</b>	<b>40.31</b>	<b>0.97</b>	<b>1.02</b>	22.05	44.575
1542M	<b>8.63</b>	<b>63.24</b>	<b>93.30</b>	<b>89.05</b>	<b>18.34</b>	<b>35.76</b>	<b>0.93</b>	<b>0.98</b>	<b>17.48</b>	42.16



*Figure 3.* Performance on the Winograd Schema Challenge as a function of model capacity.



---

# Scaling Laws for Neural Language Models

---

**Jared Kaplan \***

Johns Hopkins University, OpenAI

jaredk@jhu.edu

**Sam McCandlish\***

OpenAI

sam@openai.com

**Tom Henighan**

OpenAI

henighan@openai.com

**Tom B. Brown**

OpenAI

tom@openai.com

**Benjamin Chess**

OpenAI

bchess@openai.com

**Rewon Child**

OpenAI

rewon@openai.com

**Scott Gray**

OpenAI

scott@openai.com

**Alec Radford**

OpenAI

alec@openai.com

**Jeffrey Wu**

OpenAI

jeffwu@openai.com

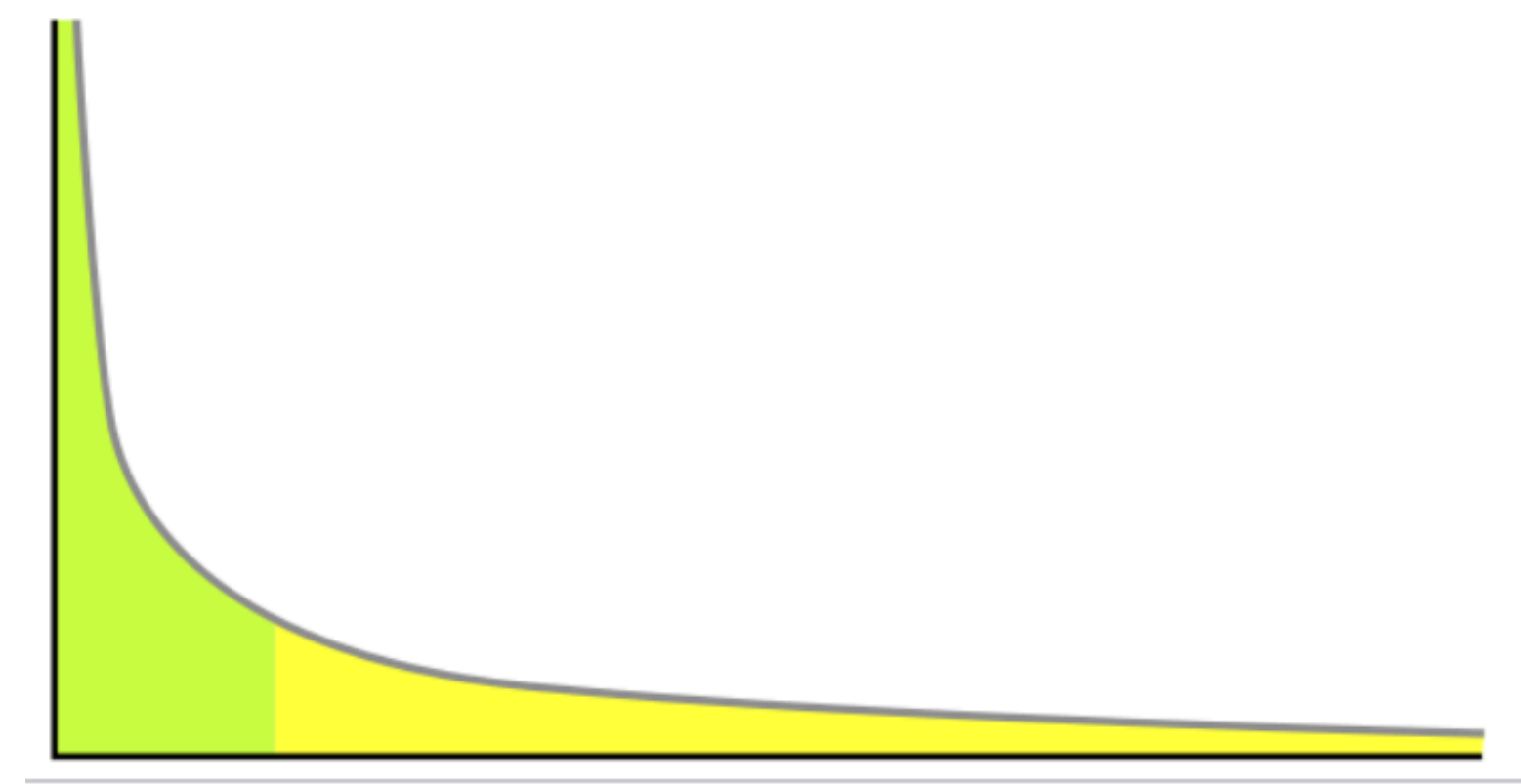
**Dario Amodei**

OpenAI

damodei@openai.com

# Scaling Laws for LLMs

## Power laws

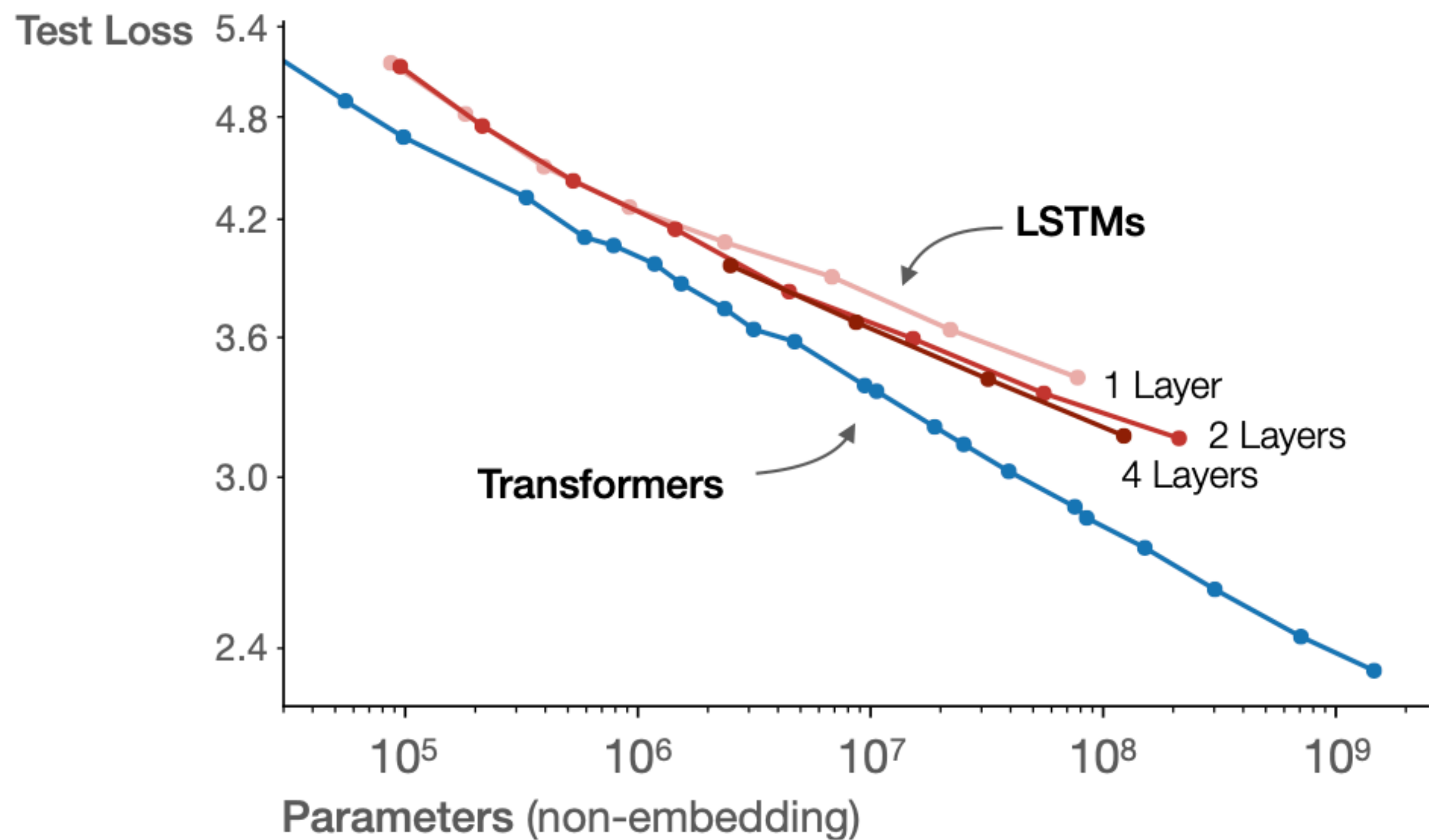


- A power law is a relation between two quantities:  $f(x) = (a/x)^k$  e.g. model performance vs. model size.
- Number of model parameters  $N$  (excluding subword embeddings)
- Size of dataset  $D$
- Amount of compute (MFLOPs)  $C$
- $N$ ,  $D$ ,  $C$  are dominant. Other choices in hyperparameters like width vs. depth are less relevant
- 1 PetaFLOP-day (PF-day) is  $8.64 \times 10^{19}$  FLOPS

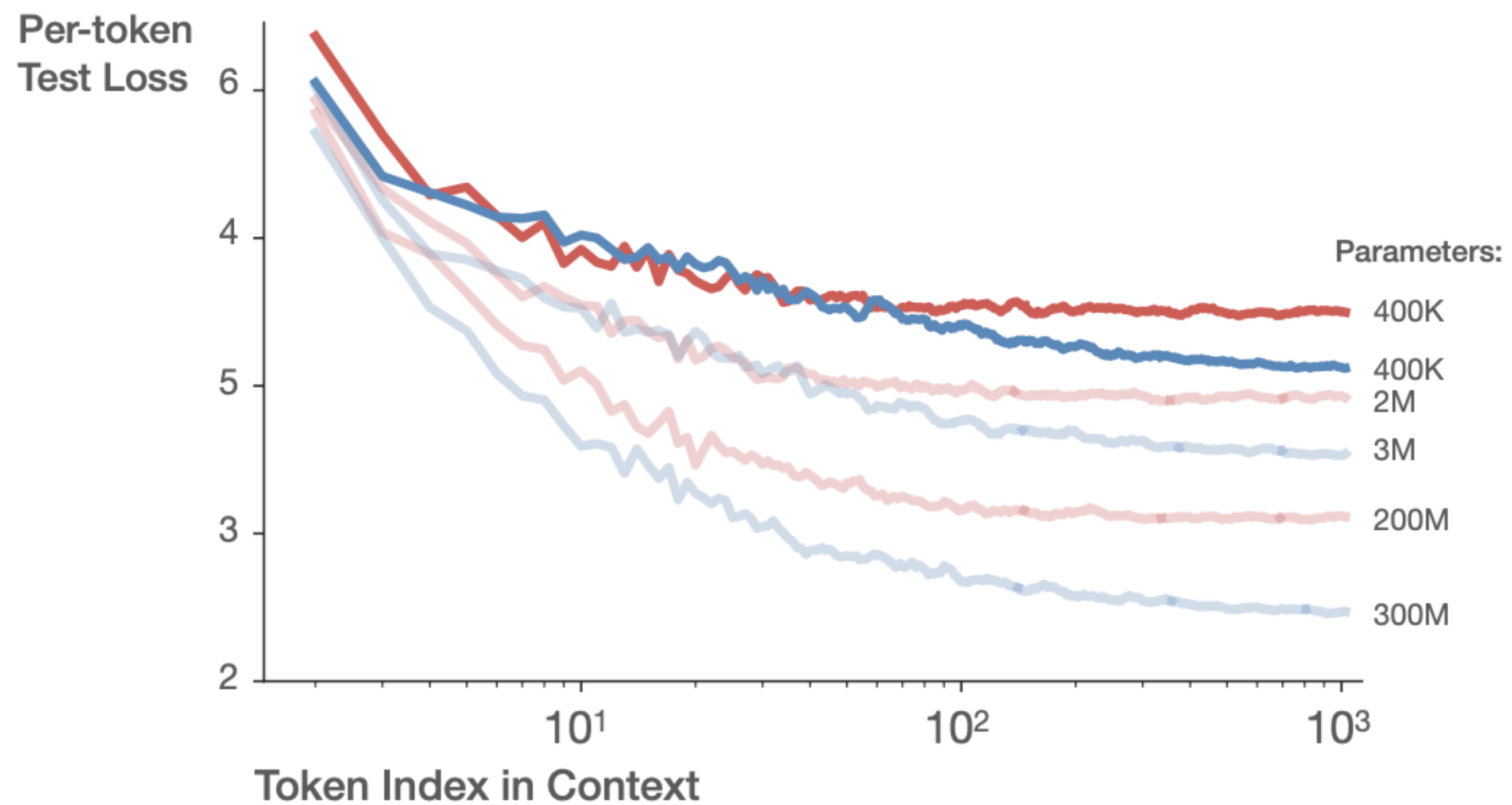
<b>Operation</b>	<b>Parameters</b>	<b>FLOPs per Token</b>
Embed	$(n_{\text{vocab}} + n_{\text{ctx}}) d_{\text{model}}$	$4d_{\text{model}}$
Attention: QKV	$n_{\text{layer}} d_{\text{model}} 3d_{\text{attn}}$	$2n_{\text{layer}} d_{\text{model}} 3d_{\text{attn}}$
Attention: Mask	—	$2n_{\text{layer}} n_{\text{ctx}} d_{\text{attn}}$
Attention: Project	$n_{\text{layer}} d_{\text{attn}} d_{\text{model}}$	$2n_{\text{layer}} d_{\text{attn}} d_{\text{embd}}$
Feedforward	$n_{\text{layer}} 2d_{\text{model}} d_{\text{ff}}$	$2n_{\text{layer}} 2d_{\text{model}} d_{\text{ff}}$
De-embed	—	$2d_{\text{model}} n_{\text{vocab}}$
<b>Total (Non-Embedding)</b>	$N = 2d_{\text{model}} n_{\text{layer}} (2d_{\text{attn}} + d_{\text{ff}})$	$C_{\text{forward}} = 2N + 2n_{\text{layer}} n_{\text{ctx}} d_{\text{attn}}$

**Table 1** Parameter counts and compute (forward pass) estimates for a Transformer model. Sub-leading terms such as nonlinearities, biases, and layer normalization are omitted.

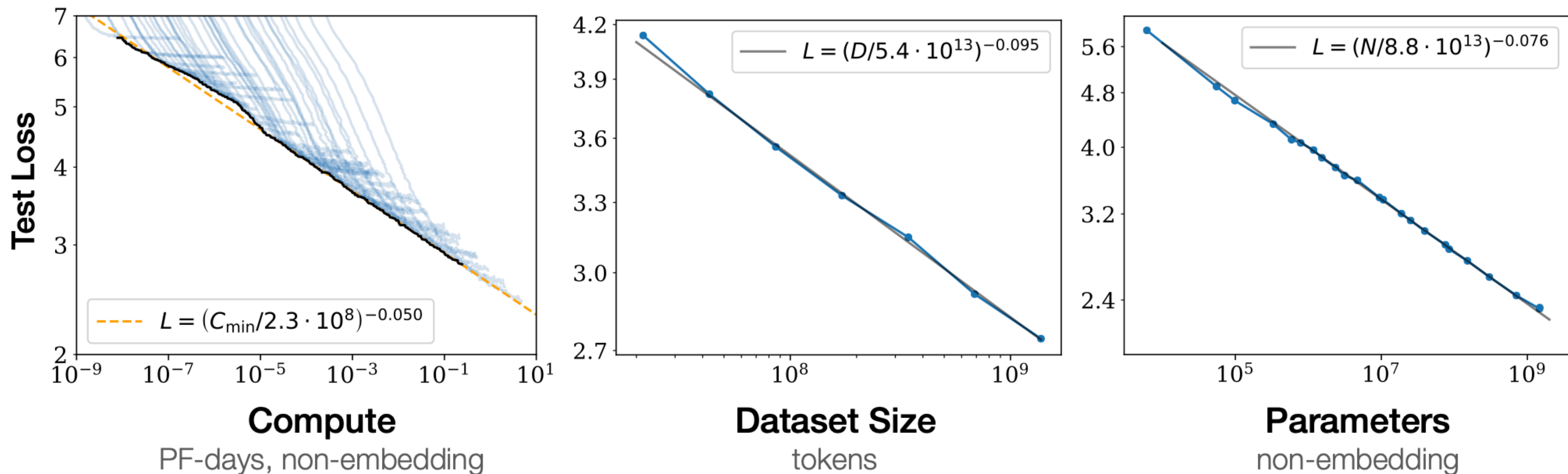
**Transformers asymptotically outperform LSTMs**  
due to improved use of long contexts



**LSTM plateaus after <100 tokens**  
**Transformer improves through the whole context**

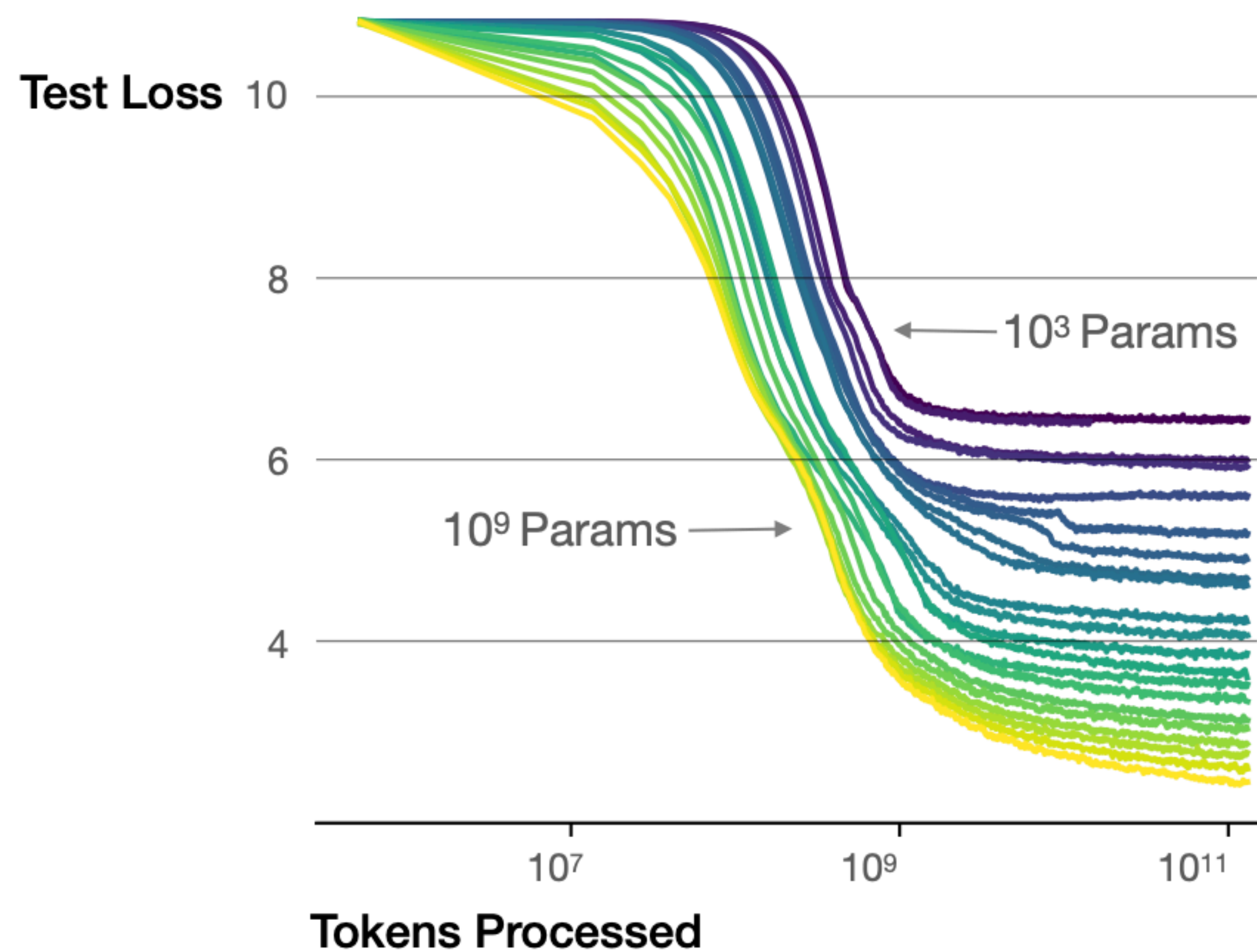




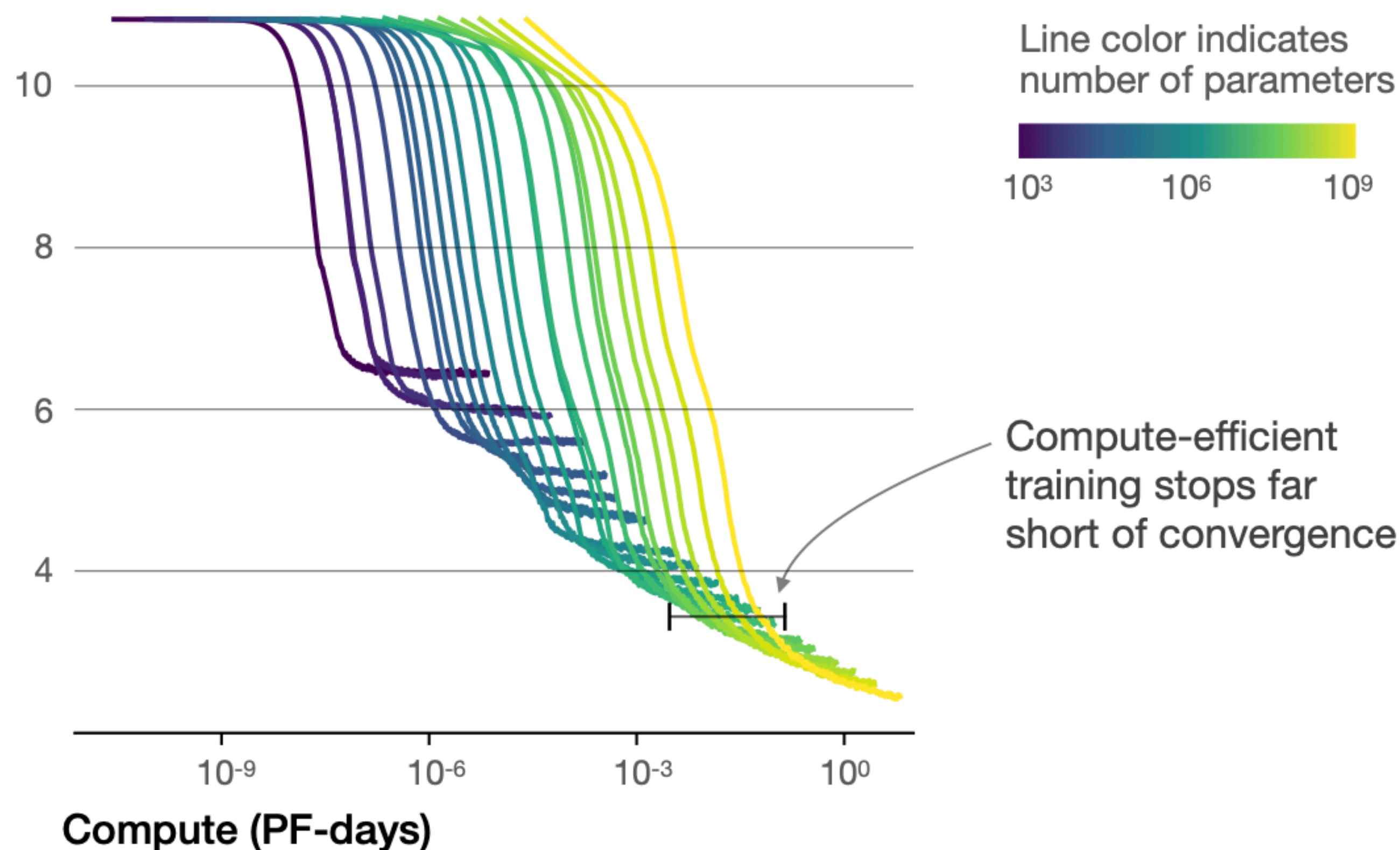


**Figure 1** Language modeling performance improves smoothly as we increase the model size, dataset size, and amount of compute<sup>2</sup> used for training. For optimal performance all three factors must be scaled up in tandem. Empirical performance has a power-law relationship with each individual factor when not bottlenecked by the other two.

Larger models require **fewer samples** to reach the same performance

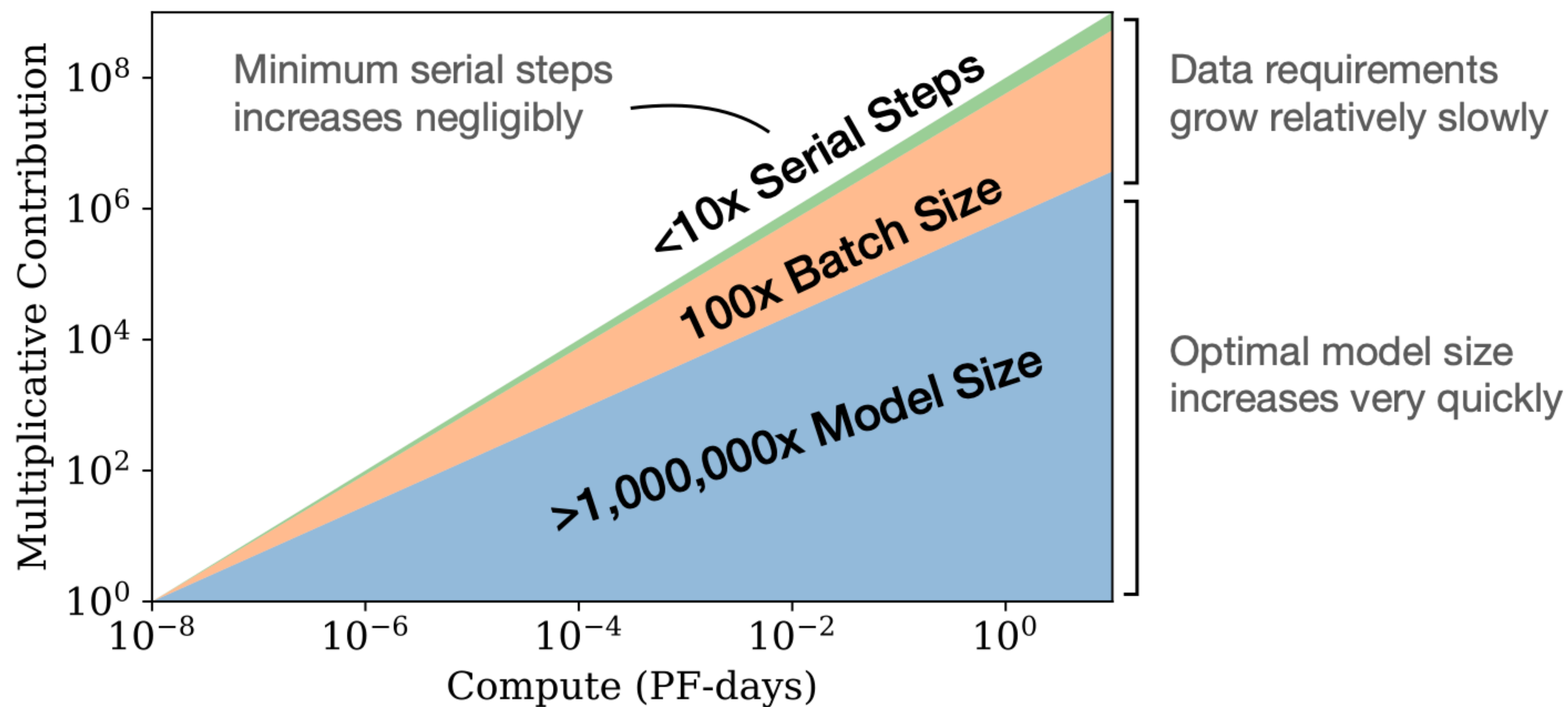


The optimal model size grows smoothly with the loss target and compute budget



**Figure 2** We show a series of language model training runs, with models ranging in size from  $10^3$  to  $10^9$  parameters (excluding embeddings).





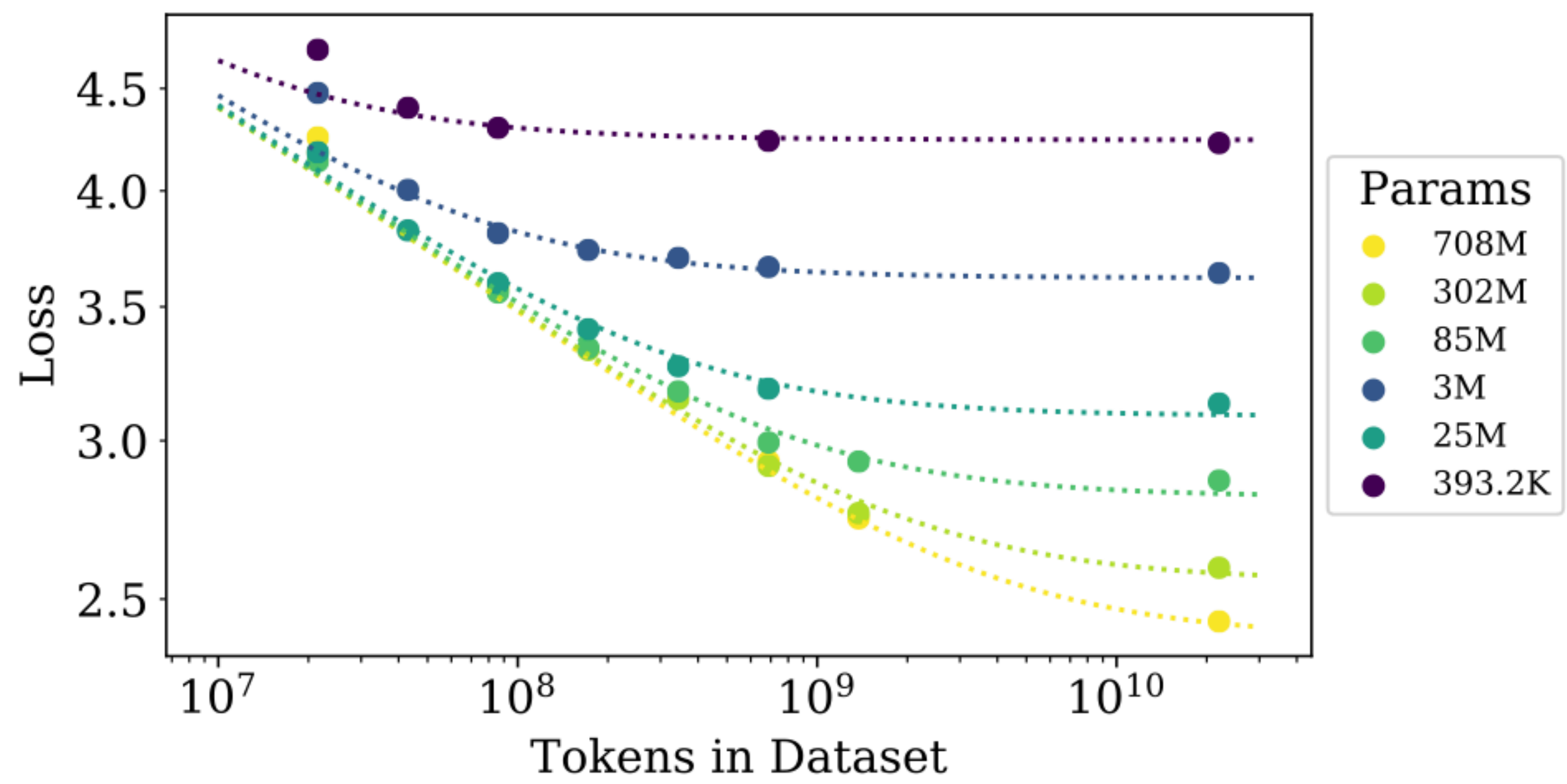
**Figure 3** As more compute becomes available, we can choose how much to allocate towards training larger models, using larger batches, and training for more steps. We illustrate this for a billion-fold increase in compute. For optimally compute-efficient training, most of the increase should go towards increased model size. A relatively small increase in data is needed to avoid reuse. Of the increase in data, most can be used to increase parallelism through larger batch sizes, with only a very small increase in serial training time required.

# Power laws for test loss

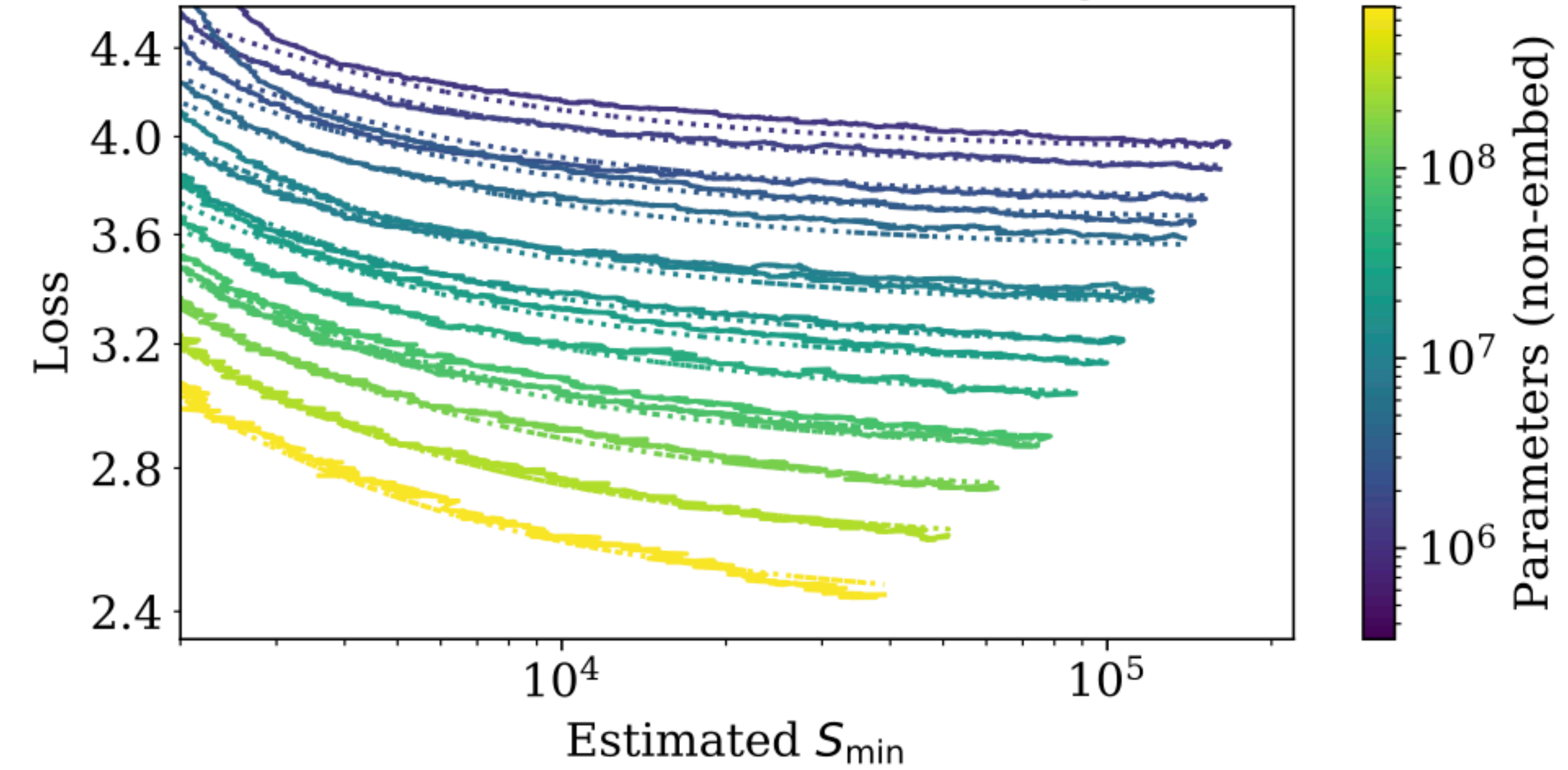
- Let  $L(\cdot)$  represent the test loss dependent on either parameters  $N$ , or dataset size  $D$  or compute  $C$
- For models with limited number of parameters:  
 $L(N) = (N_c/N)^{\alpha_N}; \alpha_N \approx 0.076, N_c \approx 8.8 \times 10^{13}$  (non-embd params)
- For models with limited dataset size:  
 $L(D) = (D_c/D)^{\alpha_D}; \alpha_D \approx 0.095, D_c \approx 5.4 \times 10^{13}$  (tokens)
- For models trained with limited compute:  
 $L(C) = (C_c^{min}/C_{min})^{\alpha_C^{min}}; \alpha_C^{min} \approx 0.050, C_c^{min} \approx 3.1 \times 10^8$  (PF-days)



Loss vs Model and Dataset Size



Loss vs Model Size and Training Steps

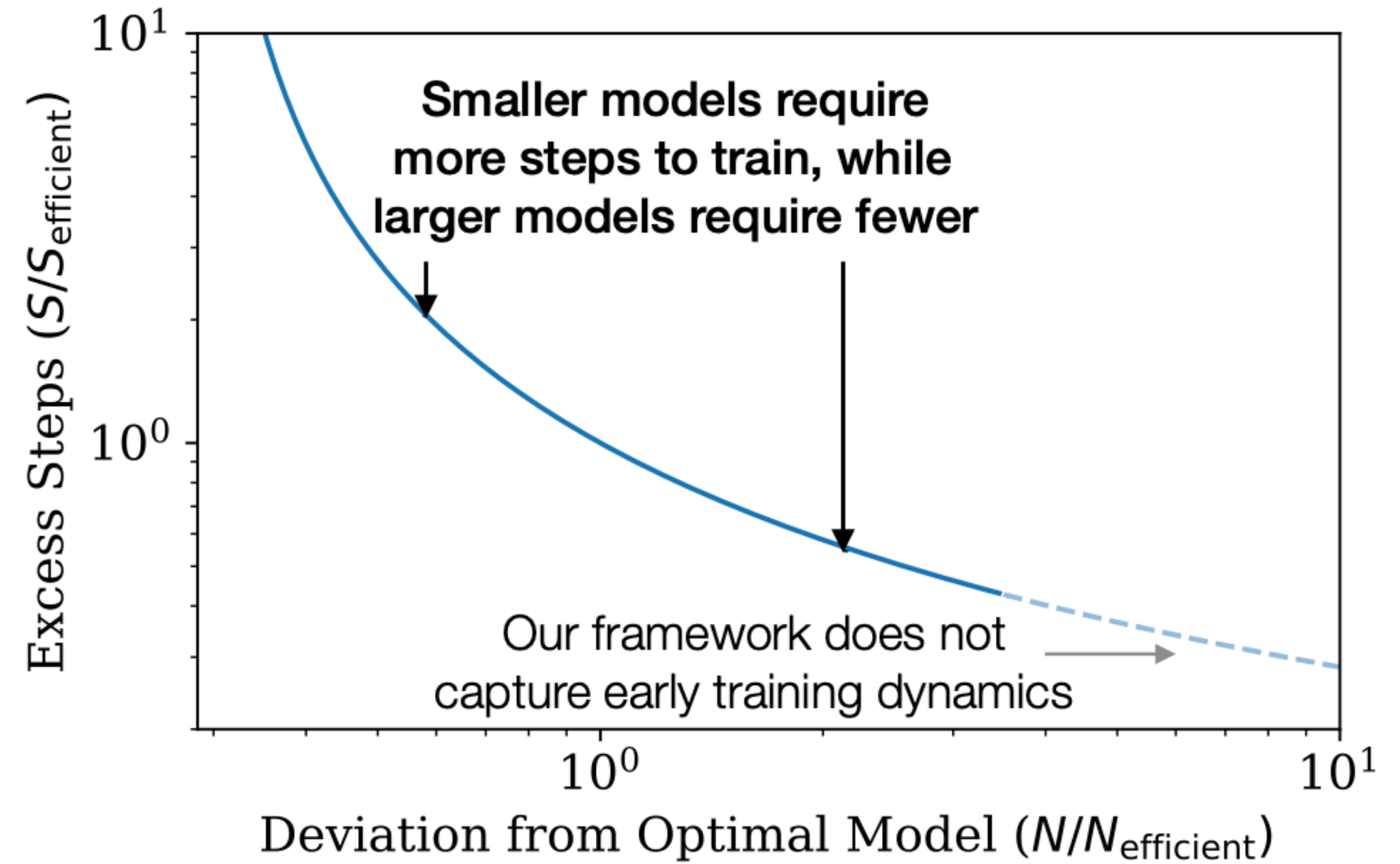
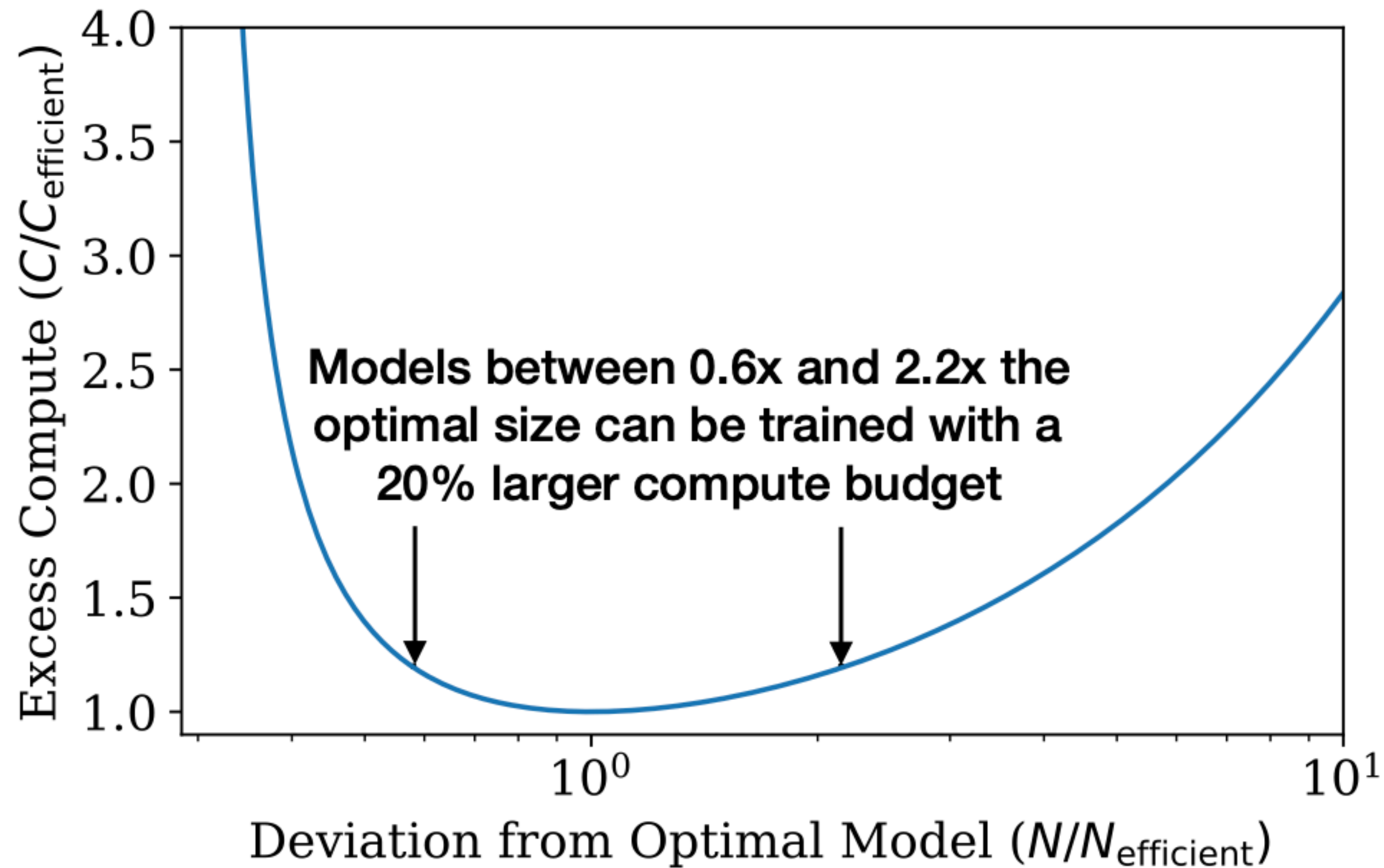


$$L(N, D) = \left[ \left( \frac{N_c}{N} \right)^{\frac{\alpha_N}{\alpha_D}} + \frac{D_c}{D} \right]^{\alpha_D}$$

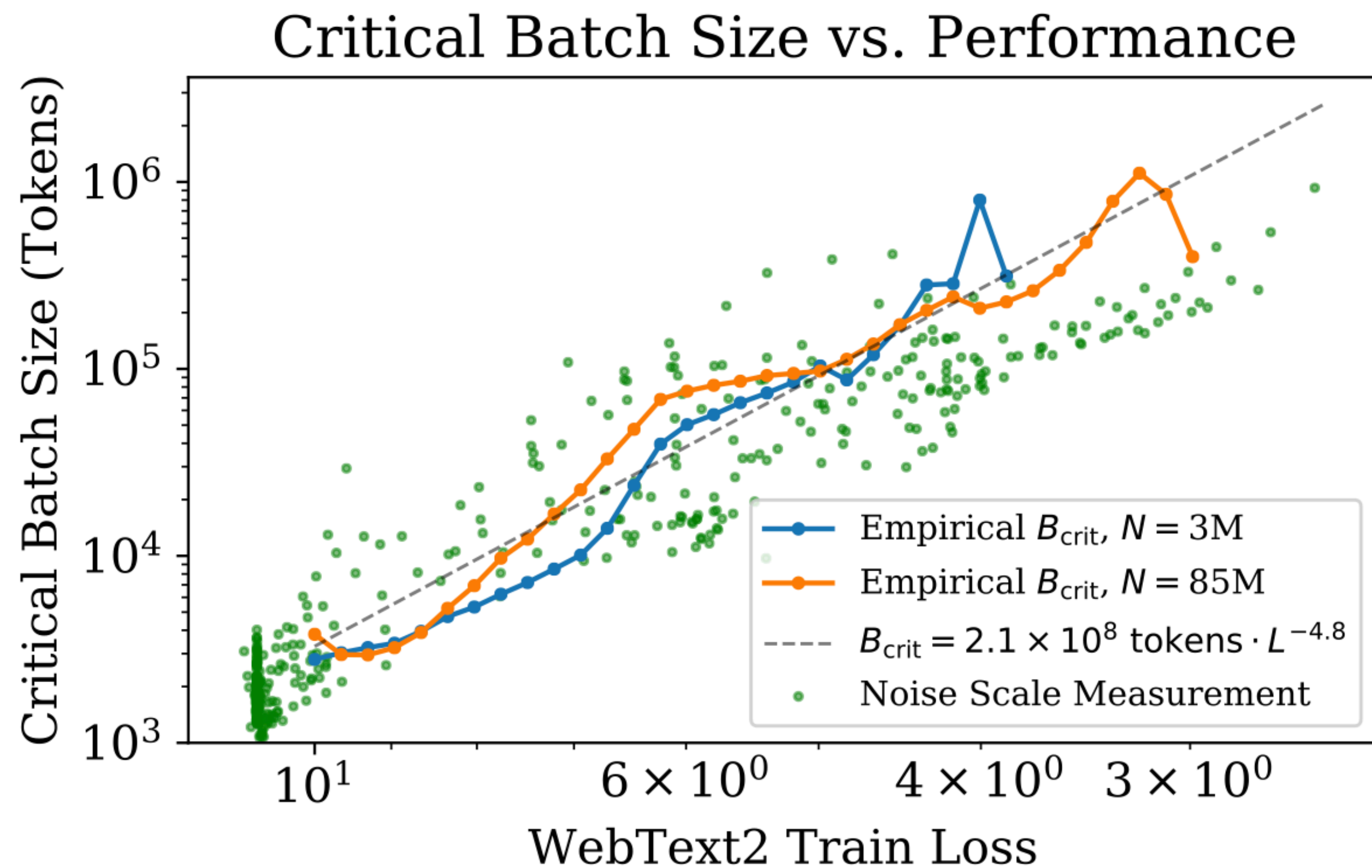
$$L(N, S) = \left( \frac{N_c}{N} \right)^{\alpha_N} + \left( \frac{S_c}{S_{\min}(S)} \right)^{\alpha_S}$$

S = parameter update steps

# Optimal Allocation of Compute Budget







**Figure 10** The critical batch size  $B_{\text{crit}}$  follows a power law in the loss as performance increase, and does not depend directly on the model size. We find that the critical batch size approximately doubles for every 13% decrease in loss.  $B_{\text{crit}}$  is measured empirically from the data shown in Figure 18, but it is also roughly predicted by the gradient noise scale, as in [MKAT18]. arXiv:1812.06162

# Lessons from scaling LLMs

- Number of model parameters  $N$   
Size of dataset  $D$
- Amount of compute (MFLOPs)  $C$
- Performance depends strongly on scale, weakly on model shape
- Performance has a power-law relationship with each of the three scale factors  $N$ ,  $D$ ,  $C$  when not bottlenecked by the other two
- Performance improves predictably as long as we scale up  $N$  and  $D$  in tandem
- Training curves follow predictable power-laws whose parameters are roughly independent of the model size

# Lessons from scaling LLMs

- Transfer to a different distribution incurs a constant penalty but otherwise improves roughly in line with performance on the training set.
- Large models are more sample-efficient than small models, reaching the same level of performance with fewer optimization steps and using fewer data points
- The ideal batch size for training these models is roughly a power of the loss only, and continues to be determinable by measuring the gradient noise scale



---

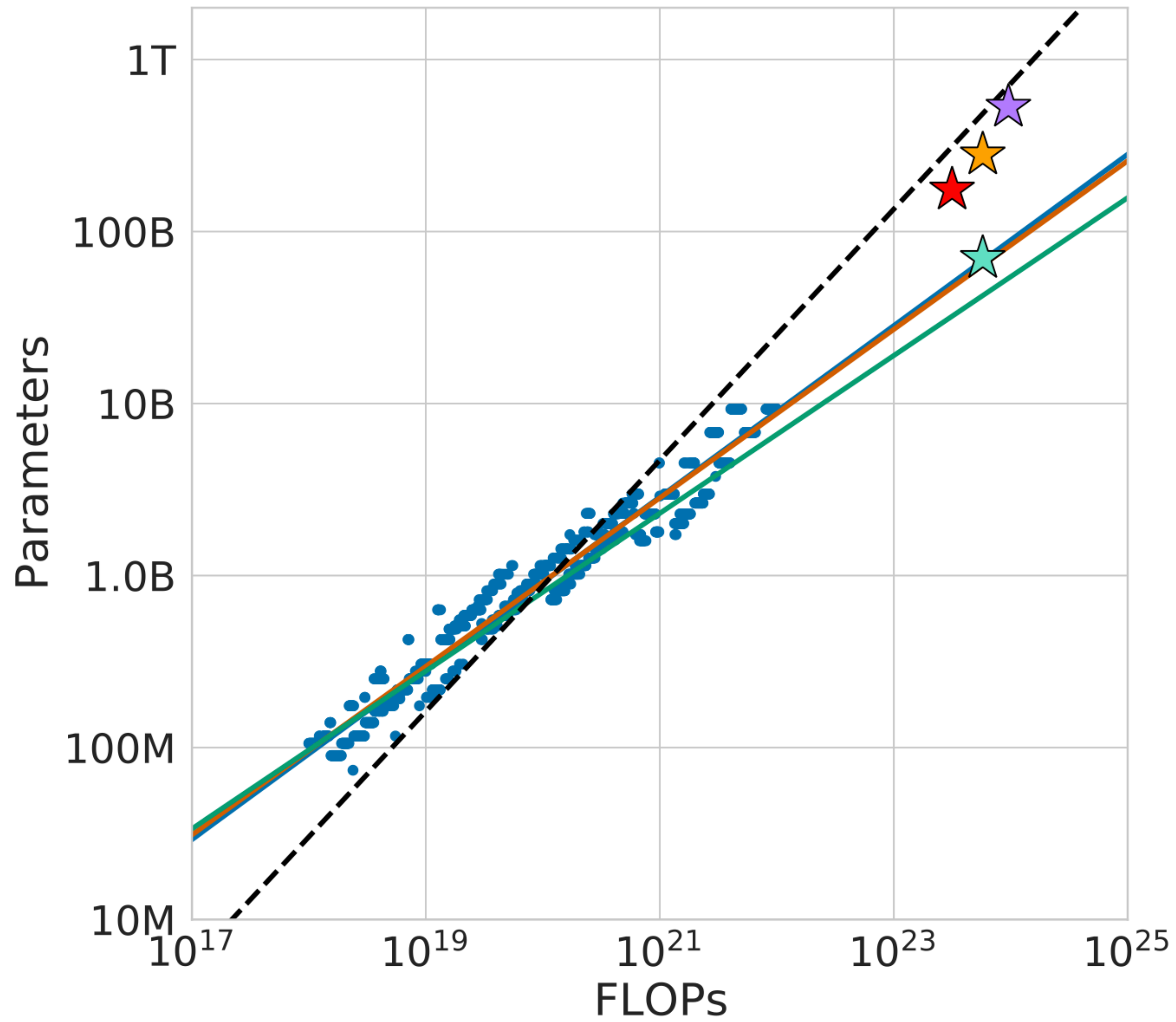
# Training Compute-Optimal Large Language Models

**Jordan Hoffmann<sup>\*</sup>, Sebastian Borgeaud<sup>\*</sup>, Arthur Mensch<sup>\*</sup>, Elena Buchatskaya, Trevor Cai, Eliza Rutherford,  
Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland,  
Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan,  
Erich Elsen, Jack W. Rae, Oriol Vinyals and Laurent Sifre<sup>\*</sup>**

# Train longer on more tokens

## Lessons from training Chinchilla

- From GPT3: large models should not be trained to lowest possible loss to be compute optimal
- Question: **Given a fixed FLOPs budget how should one trade off model size and number of training tokens?**
- Pre-training loss  $L(N, D)$  for  $N$  parameters and  $D$  training tokens. Find the optimal  $N$  and  $D$  values for a given compute budget.
- Empirical study on training 400 models from 70M to 16B parameters, trained on 5B to 400B tokens.
- Answer: **Train smaller models for (a lot) more training steps.**



- Approach 1
- Approach 2
- Approach 3
- Kaplan et al (2020)
- Chinchilla (70B)
- Gopher (280B)
- GPT-3 (175B)
- Megatron-Turing NLG (530B)



---

Model	Size (# Parameters)	Training Tokens
LaMDA ( <a href="#">Thoppilan et al., 2022</a> )	137 Billion	168 Billion
GPT-3 ( <a href="#">Brown et al., 2020</a> )	175 Billion	300 Billion
Jurassic ( <a href="#">Lieber et al., 2021</a> )	178 Billion	300 Billion
<i>Gopher</i> ( <a href="#">Rae et al., 2021</a> )	280 Billion	300 Billion
MT-NLG 530B ( <a href="#">Smith et al., 2022</a> )	530 Billion	270 Billion
<i>Chinchilla</i>	70 Billion	1.4 Trillion

---

---

# Language Models are Few-Shot Learners

---

**Tom B. Brown\***

**Benjamin Mann\***

**Nick Ryder\***

**Melanie Subbiah\***

**Jared Kaplan<sup>†</sup>**

**Prafulla Dhariwal**

**Arvind Neelakantan**

**Pranav Shyam**

**Girish Sastry**

**Amanda Askell**

**Sandhini Agarwal**

**Ariel Herbert-Voss**

**Gretchen Krueger**

**Tom Henighan**

**Rewon Child**

**Aditya Ramesh**

**Daniel M. Ziegler**

**Jeffrey Wu**

**Clemens Winter**

**Christopher Hesse**

**Mark Chen**

**Eric Sigler**

**Mateusz Litwin**

**Scott Gray**

**Benjamin Chess**

**Jack Clark**

**Christopher Berner**

**Sam McCandlish**

**Alec Radford**

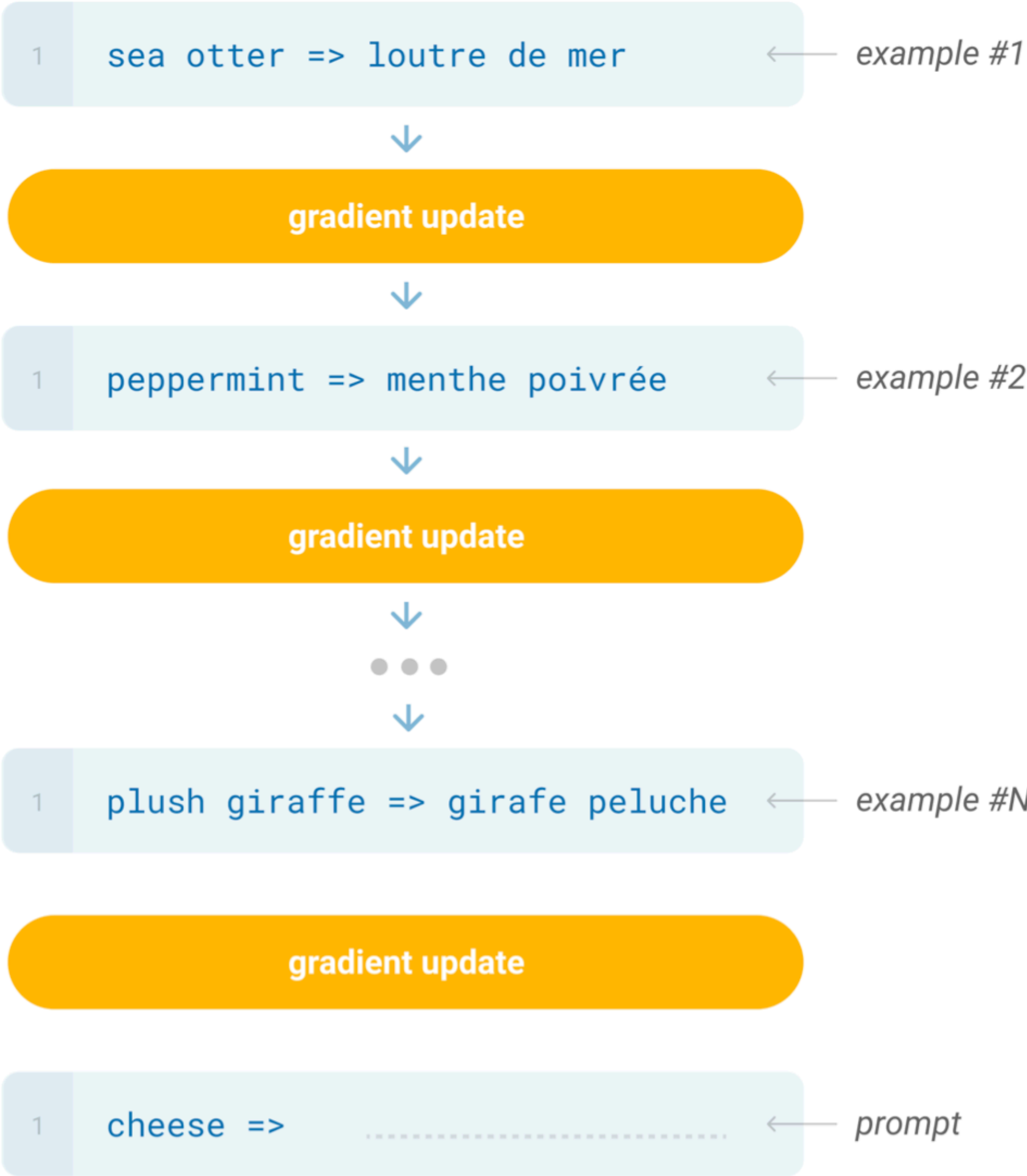
**Ilya Sutskever**

**Dario Amodei**

# Traditional fine-tuning (not used for GPT-3)

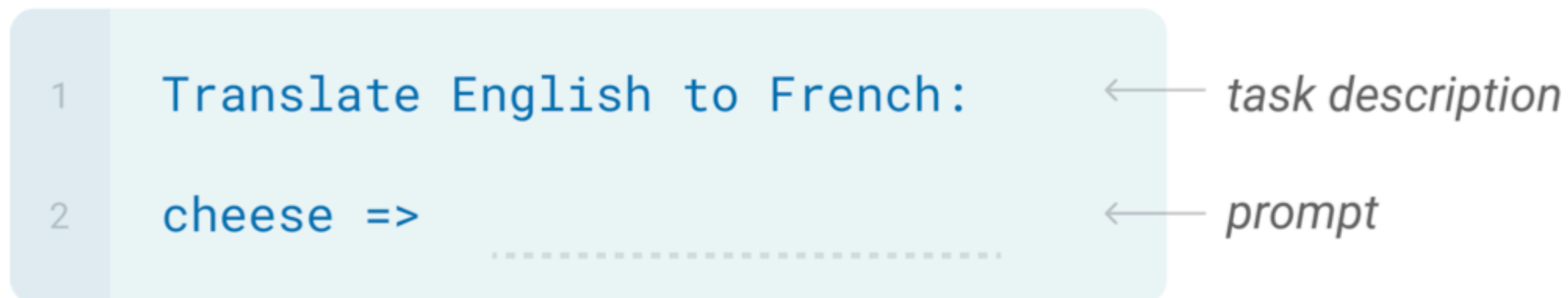
## Fine-tuning

The model is trained via repeated gradient updates using a large corpus of example tasks.



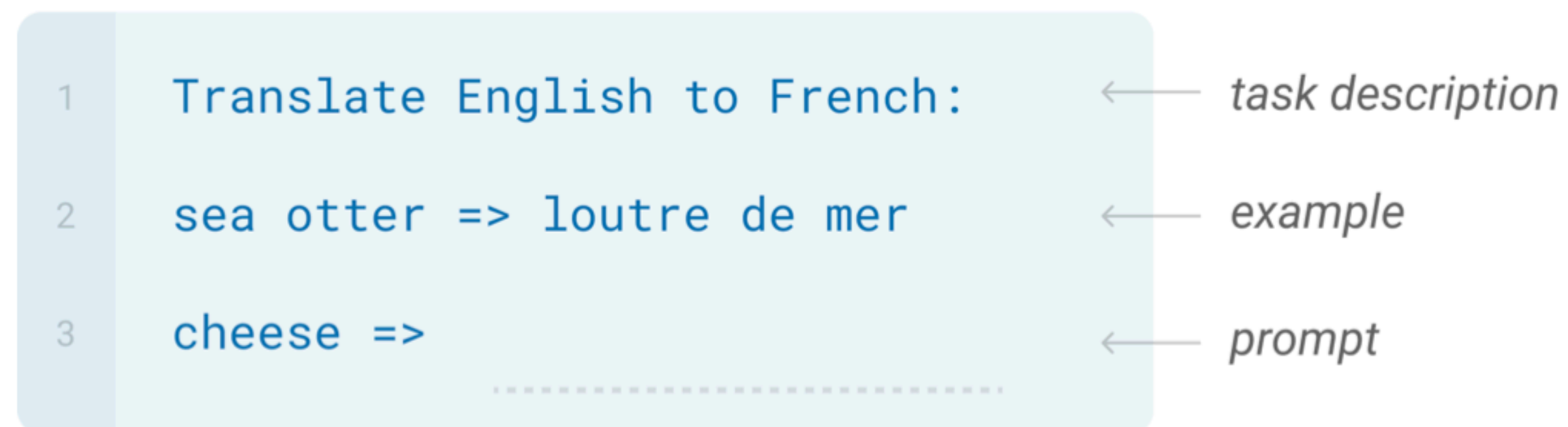
## Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.



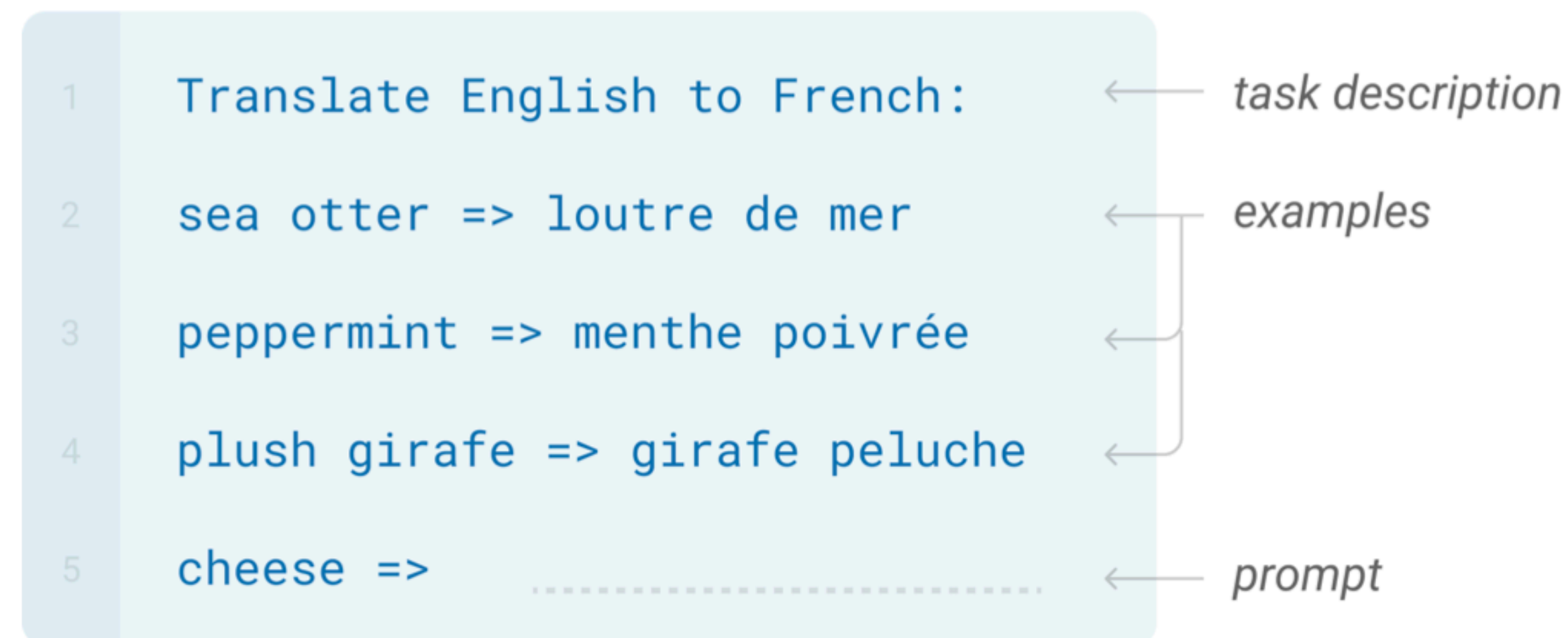
## One-shot

In addition to the task description, the model sees a single example of the task. No gradient updates are performed.



## Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.



# Fine-tuning fails at scale

- LLMs  $>10B$  parameters are very difficult to fine-tune and requires a big compute budget
- So in-context learning using a long prompt or prefix is needed to coax the answer from a "predict the next token" approach to solving multiple tasks
- Pre-training on web-scale text can observe many different tasks in-context during training in the inner loop (per batch)
- Gradient descent improves the model representations based on next token prediction over many batch updates in the outer loop



outer loop

Learning via SGD during unsupervised pre-training

inner loop

1	5 + 8 = 13
2	7 + 2 = 9
3	1 + 0 = 1
4	3 + 4 = 7
5	5 + 9 = 14
6	9 + 8 = 17

↑  
sequence #1

In-context learning

1	gaot => goat
2	sakne => snake
3	brid => bird
4	fsih => fish
5	dcuk => duck
6	cmihp => chimp

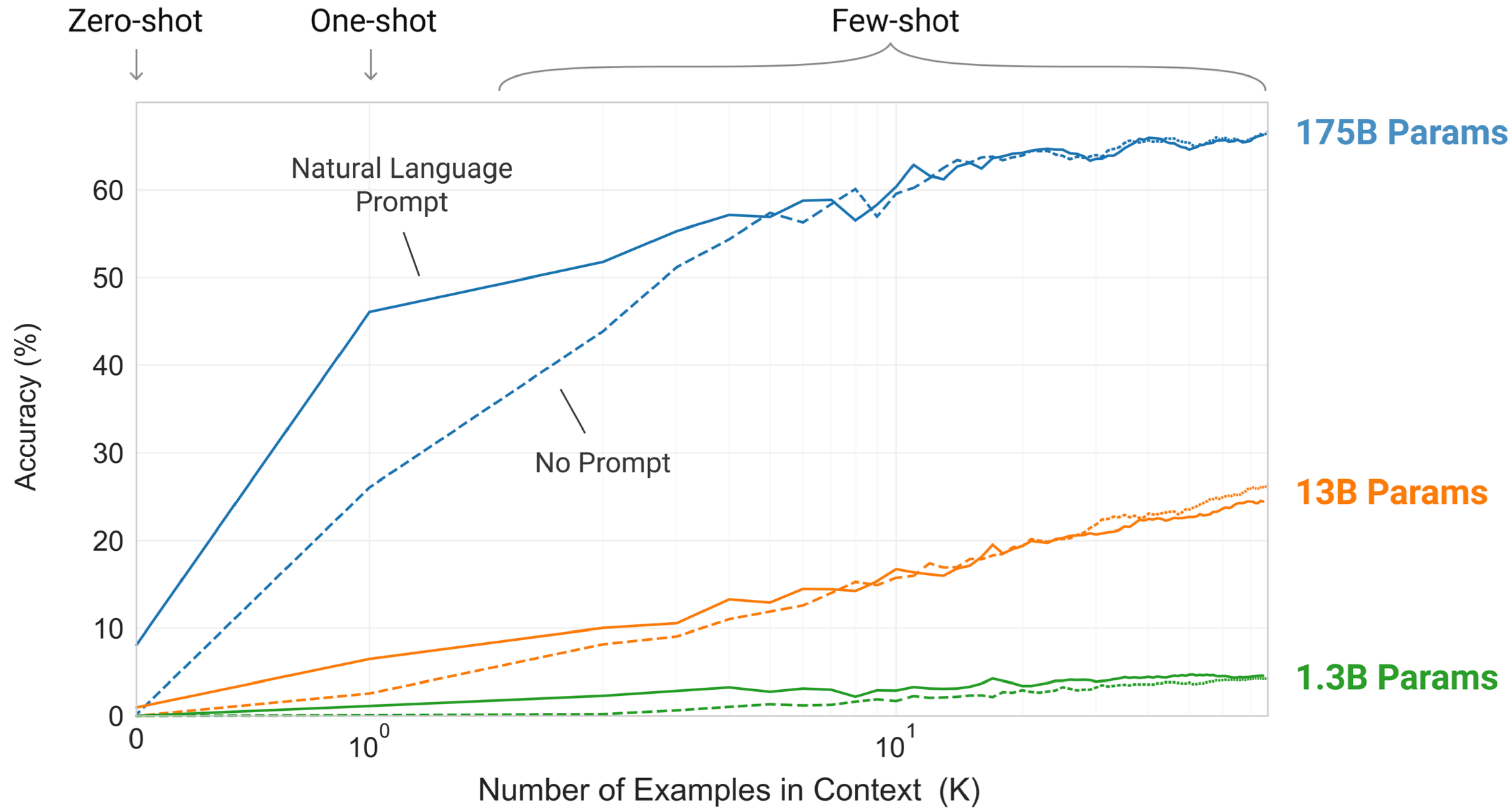
↑  
sequence #2

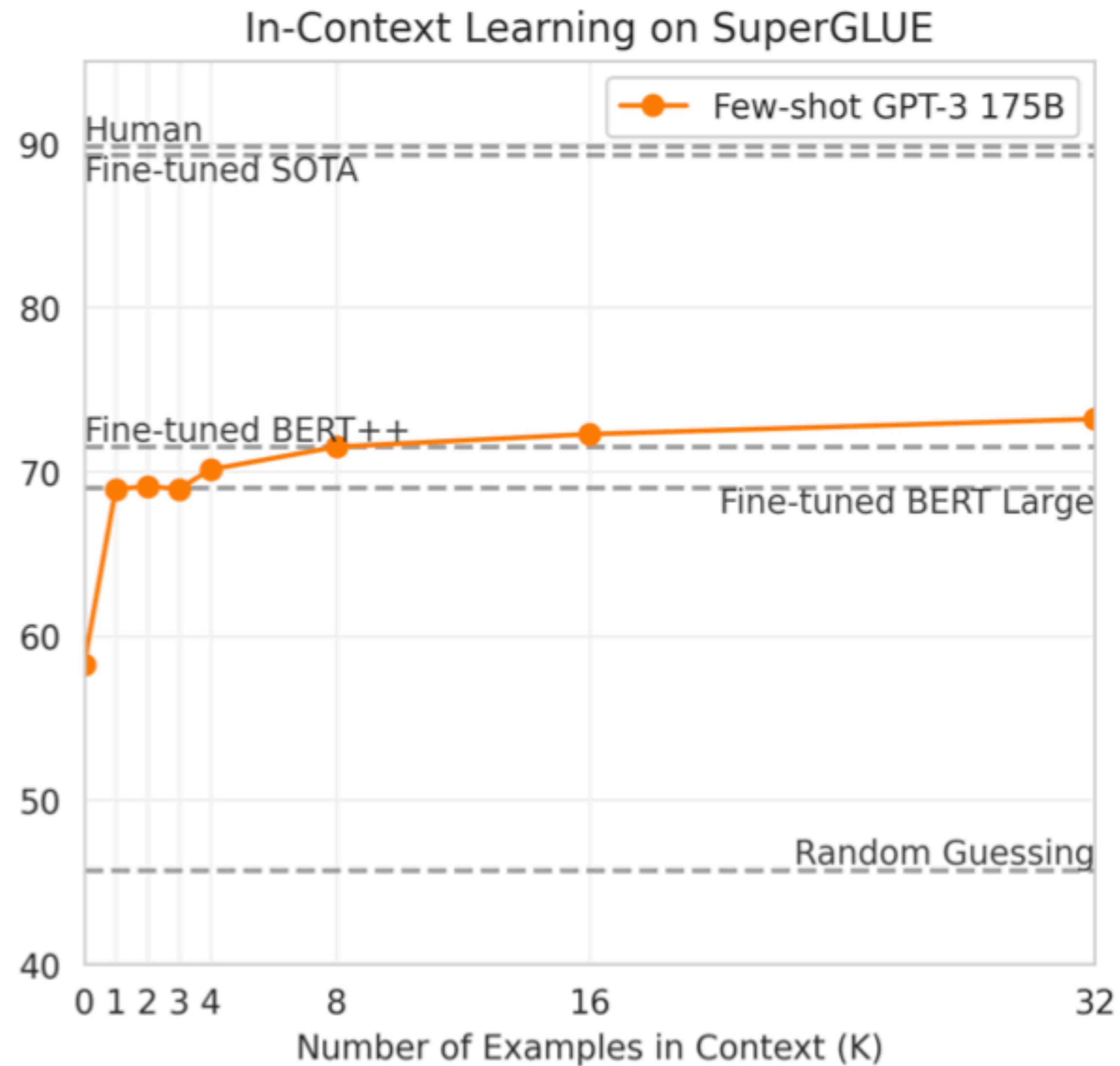
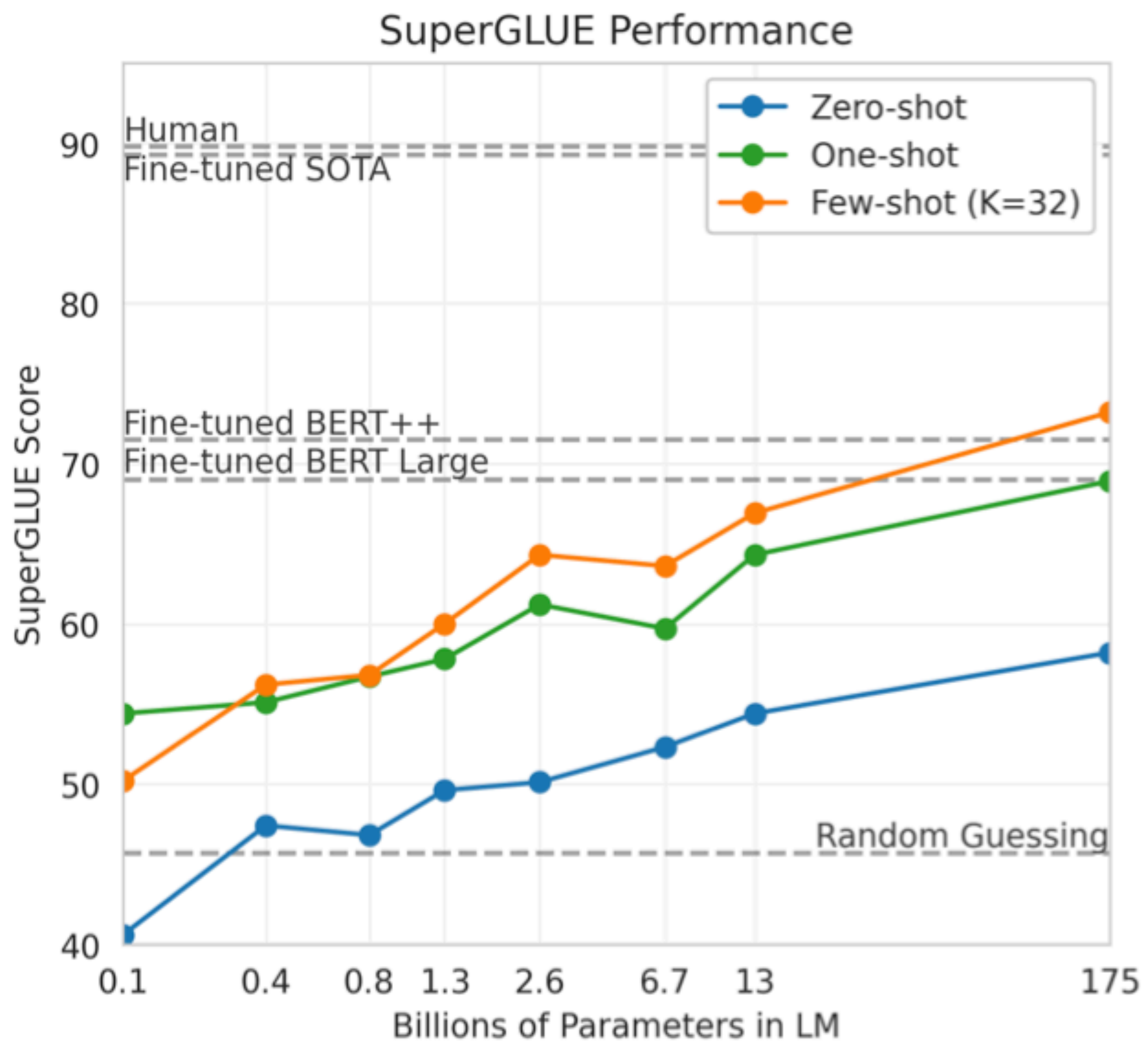
In-context learning

1	thanks => merci
2	hello => bonjour
3	mint => menthe
4	wall => mur
5	otter => loutre
6	bread => pain

↑  
sequence #3

In-context learning





**Performance on SuperGLUE increases with number of examples in context.** We find the difference in performance between the BERT-Large and BERT++ to be roughly equivalent to the difference between GPT-3 with one example per context versus eight examples per context.



	SuperGLUE Average	BoolQ Accuracy	CB Accuracy	CB F1	COPA Accuracy	RTE Accuracy
Fine-tuned SOTA	<b>89.0</b>	<b>91.0</b>	<b>96.9</b>	<b>93.9</b>	<b>94.8</b>	<b>92.5</b>
Fine-tuned BERT-Large	69.0	77.4	83.6	75.7	70.6	71.7
GPT-3 Few-Shot	71.8	76.4	75.6	52.0	92.0	69.0

	WiC Accuracy	WSC Accuracy	MultiRC Accuracy	MultiRC F1a	ReCoRD Accuracy	ReCoRD F1
Fine-tuned SOTA	<b>76.1</b>	<b>93.8</b>	<b>62.3</b>	<b>88.2</b>	<b>92.5</b>	<b>93.3</b>
Fine-tuned BERT-Large	69.6	64.6	24.1	70.0	71.3	72.0
GPT-3 Few-Shot	49.4	80.1	30.5	75.4	90.2	91.1

**Table 3.5:** Performance of GPT-3 on SuperGLUE compared to fine-tuned baselines and SOTA. All results are reported on the test set. GPT-3 few-shot is given a total of 32 examples within the context of each task and performs no gradient updates.

Setting	LAMBADA (acc)	LAMBADA (ppl)	StoryCloze (acc)	HellaSwag (acc)
SOTA	68.0 <sup>a</sup>	8.63 <sup>b</sup>	<b>91.8<sup>c</sup></b>	<b>85.6<sup>d</sup></b>
GPT-3 Zero-Shot	<b>76.2</b>	<b>3.00</b>	83.2	78.9
GPT-3 One-Shot	<b>72.5</b>	<b>3.35</b>	84.7	78.1
GPT-3 Few-Shot	<b>86.4</b>	<b>1.92</b>	87.7	79.3

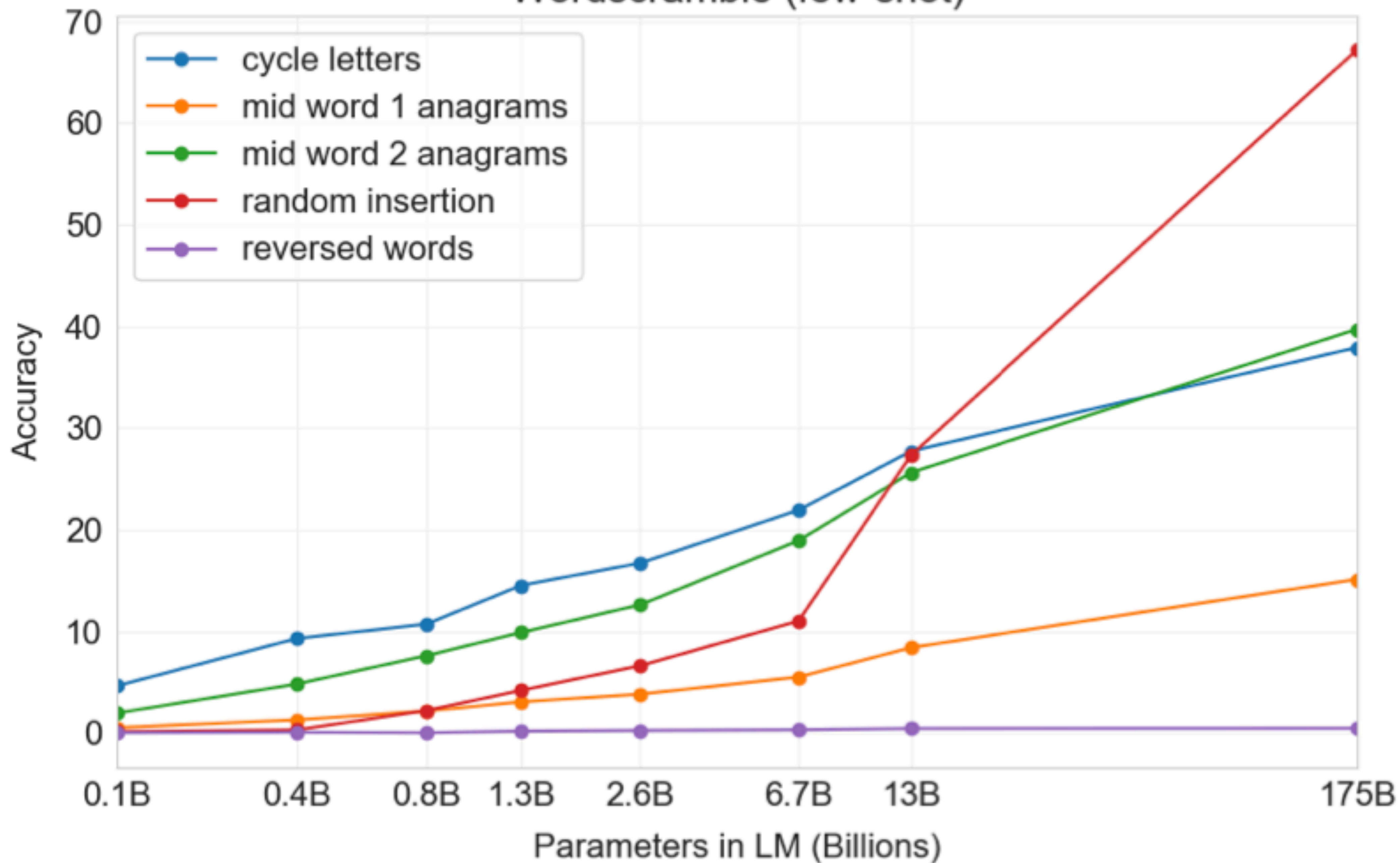
Setting	NaturalQS	WebQS	TriviaQA
RAG (Fine-tuned, Open-Domain) [LPP <sup>+</sup> 20]	<b>44.5</b>	<b>45.5</b>	<b>68.0</b>
T5-11B+SSM (Fine-tuned, Closed-Book) [RRS20]	36.6	44.7	60.5
T5-11B (Fine-tuned, Closed-Book)	34.5	37.4	50.1
GPT-3 Zero-Shot	14.6	14.4	64.3
GPT-3 One-Shot	23.0	25.3	<b>68.0</b>
GPT-3 Few-Shot	29.9	41.5	<b>71.2</b>

Setting	ARC (Easy)	ARC (Challenge)	CoQA	DROP
Fine-tuned SOTA	<b>92.0<sup>a</sup></b>	<b>78.5<sup>b</sup></b>	<b>90.7<sup>c</sup></b>	<b>89.1<sup>d</sup></b>
GPT-3 Zero-Shot	68.8	51.4	81.5	23.6
GPT-3 One-Shot	71.2	53.2	84.0	34.3
GPT-3 Few-Shot	70.1	51.5	85.0	36.5

Setting	En→Fr	Fr→En	En→De	De→En	En→Ro	Ro→En
SOTA (Supervised)	<b>45.6<sup>a</sup></b>	35.0 <sup>b</sup>	<b>41.2<sup>c</sup></b>	40.2 <sup>d</sup>	<b>38.5<sup>e</sup></b>	<b>39.9<sup>e</sup></b>
XLM [LC19]	33.4	33.3	26.4	34.3	33.3	31.8
MASS [STQ <sup>+</sup> 19]	<u>37.5</u>	34.9	28.3	35.2	<u>35.2</u>	33.1
mBART [LGG <sup>+</sup> 20]	-	-	<u>29.8</u>	34.0	35.0	30.5
GPT-3 Zero-Shot	25.2	21.2	24.6	27.2	14.1	19.9
GPT-3 One-Shot	28.3	33.7	26.2	30.4	20.6	38.6
GPT-3 Few-Shot	32.6	<u>39.2</u>	29.7	<u>40.6</u>	21.0	<u>39.5</u>

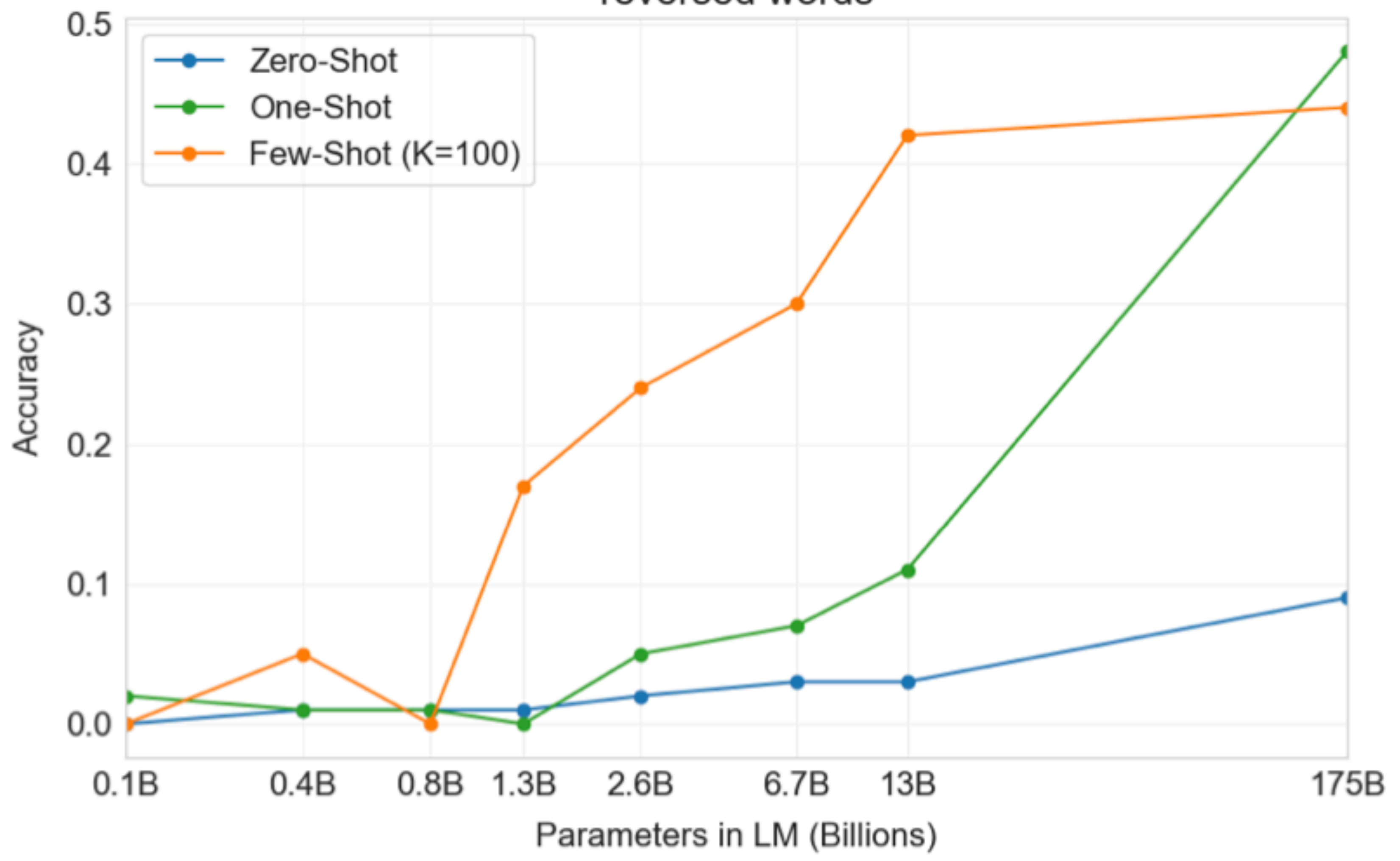


### Wordscramble (few-shot)

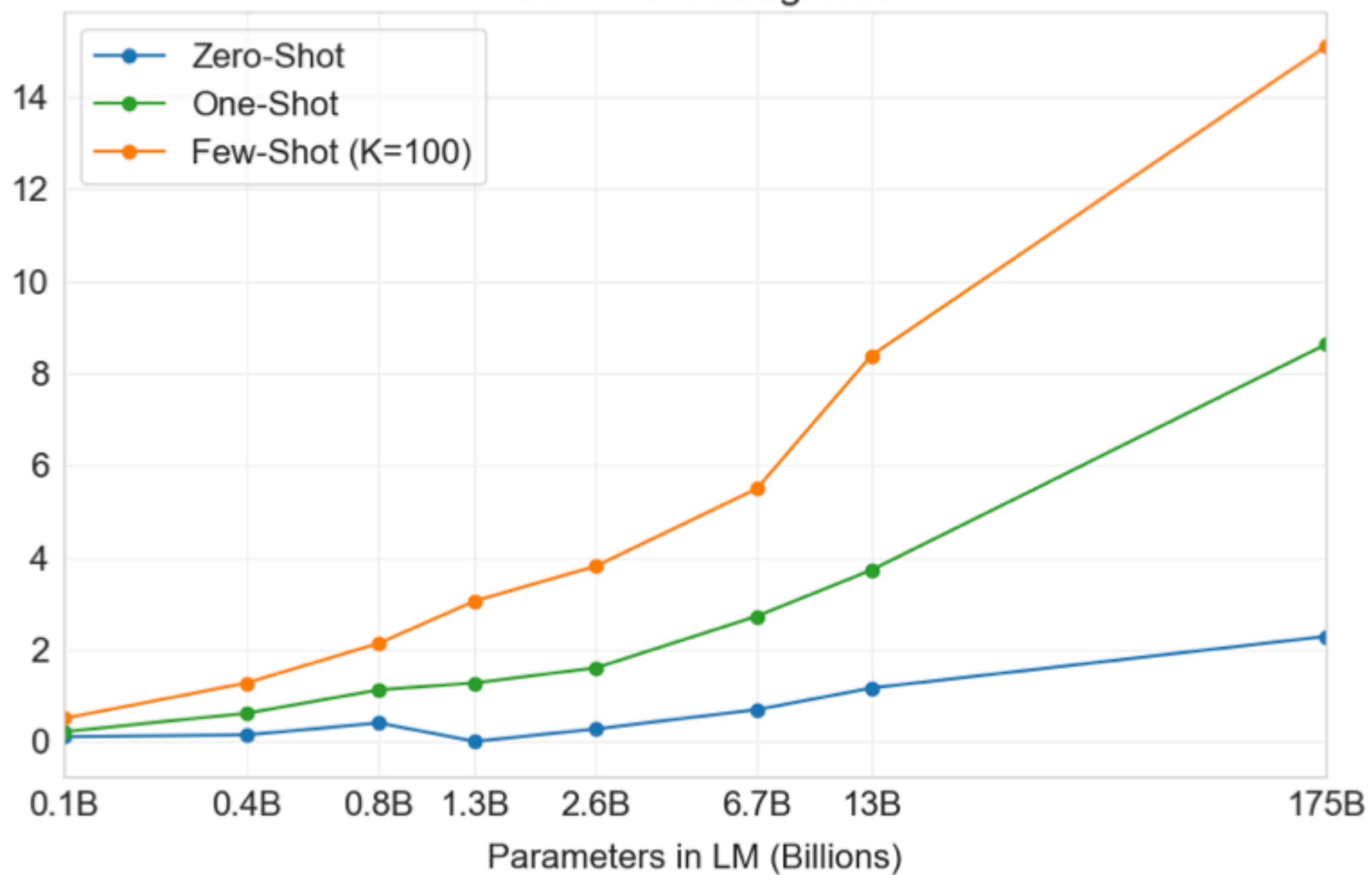




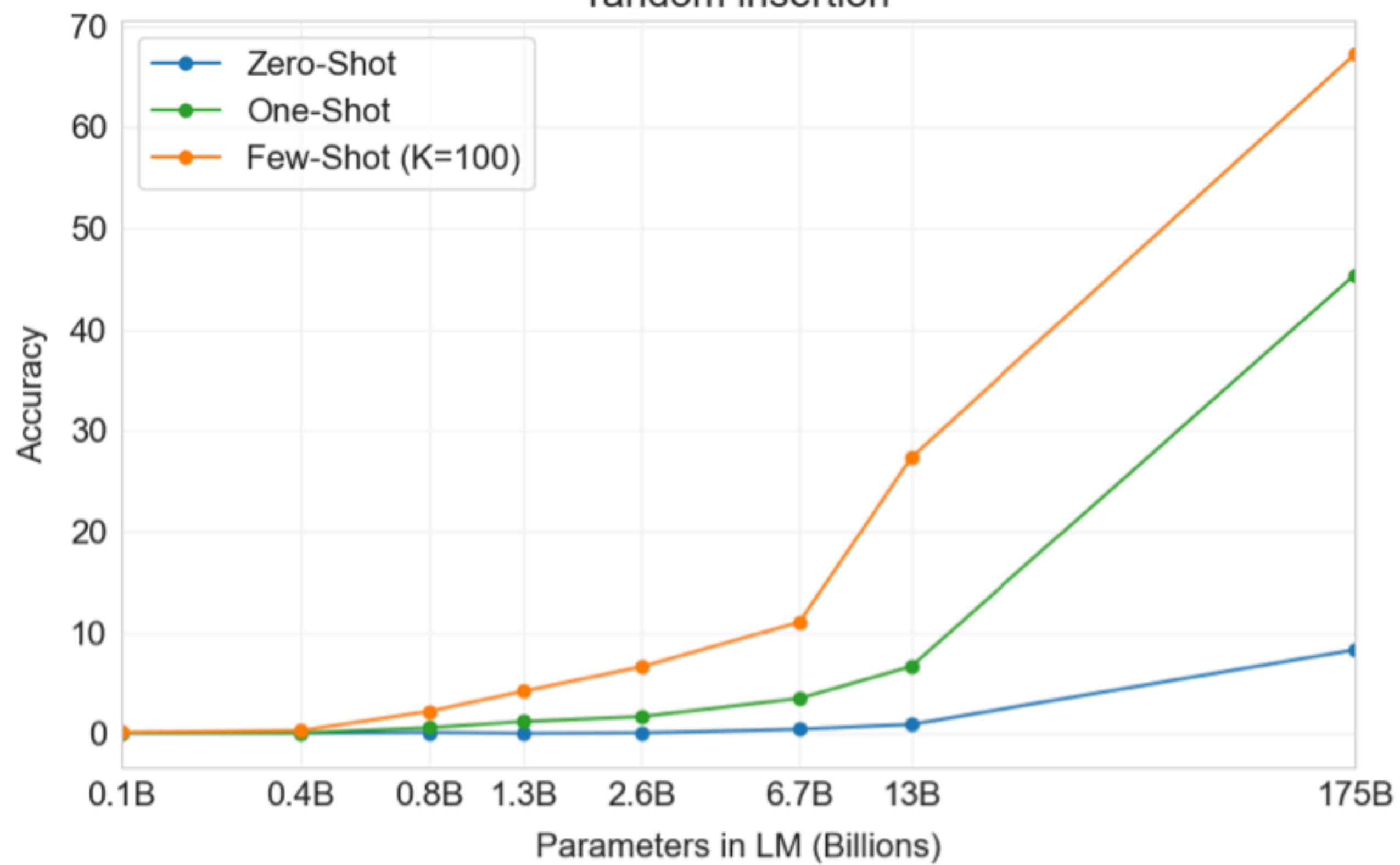
reversed words

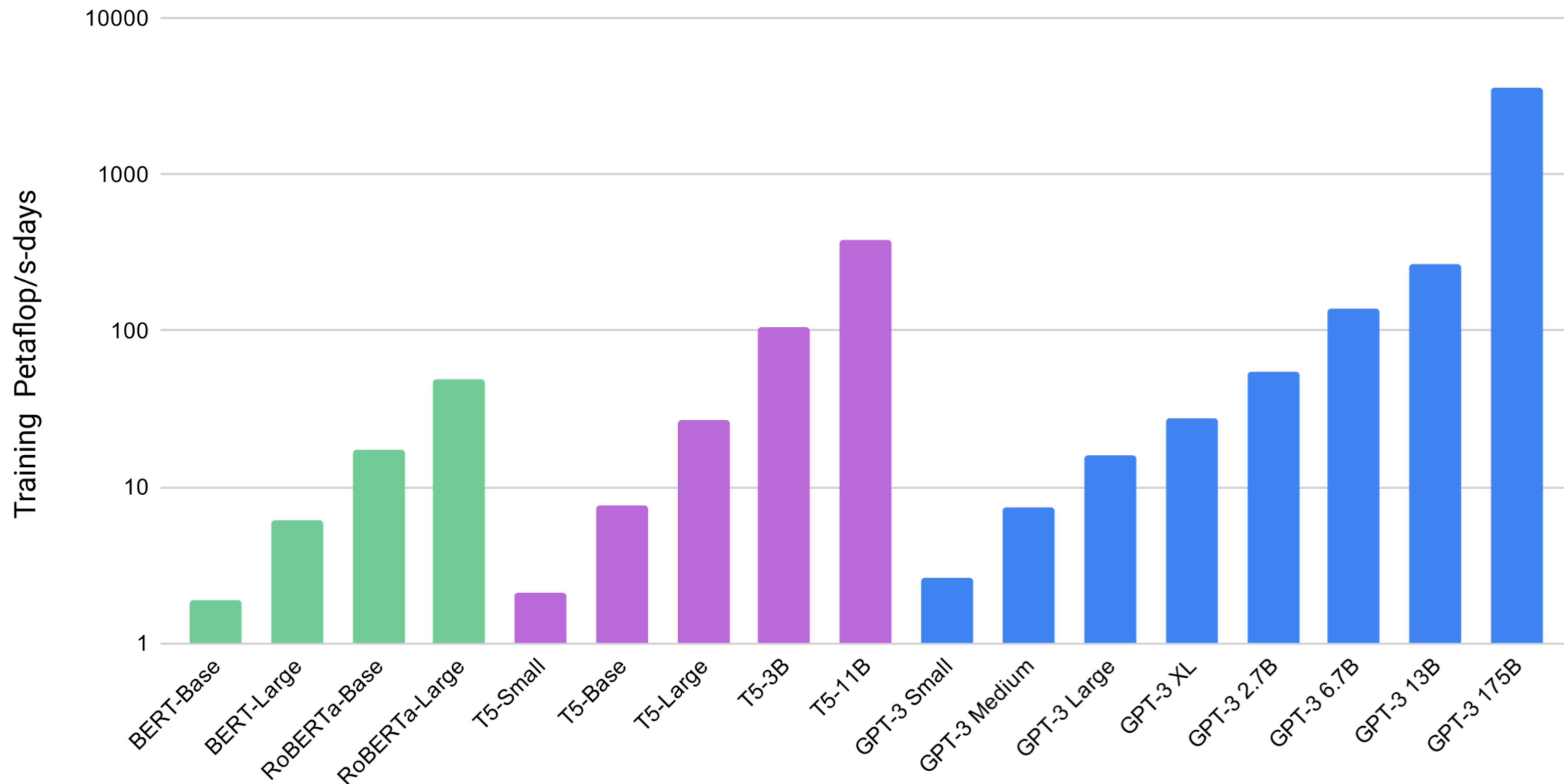


mid word 1 anagrams



random insertion





**Figure 7.2: Total compute used during training.** Based on the analysis in Scaling Laws For Neural Language Models [KMH<sup>+</sup>20] we train much larger models on many fewer tokens than is typical. As a consequence, although GPT-3 3B is almost 10x larger than RoBERTa-Large (355M params), both models took roughly 50 petaflop/s-days of compute during pre-training. Methodology for these calculations can be found in the Appendix.

---

# GLaM: Efficient Scaling of Language Models with Mixture-of-Experts

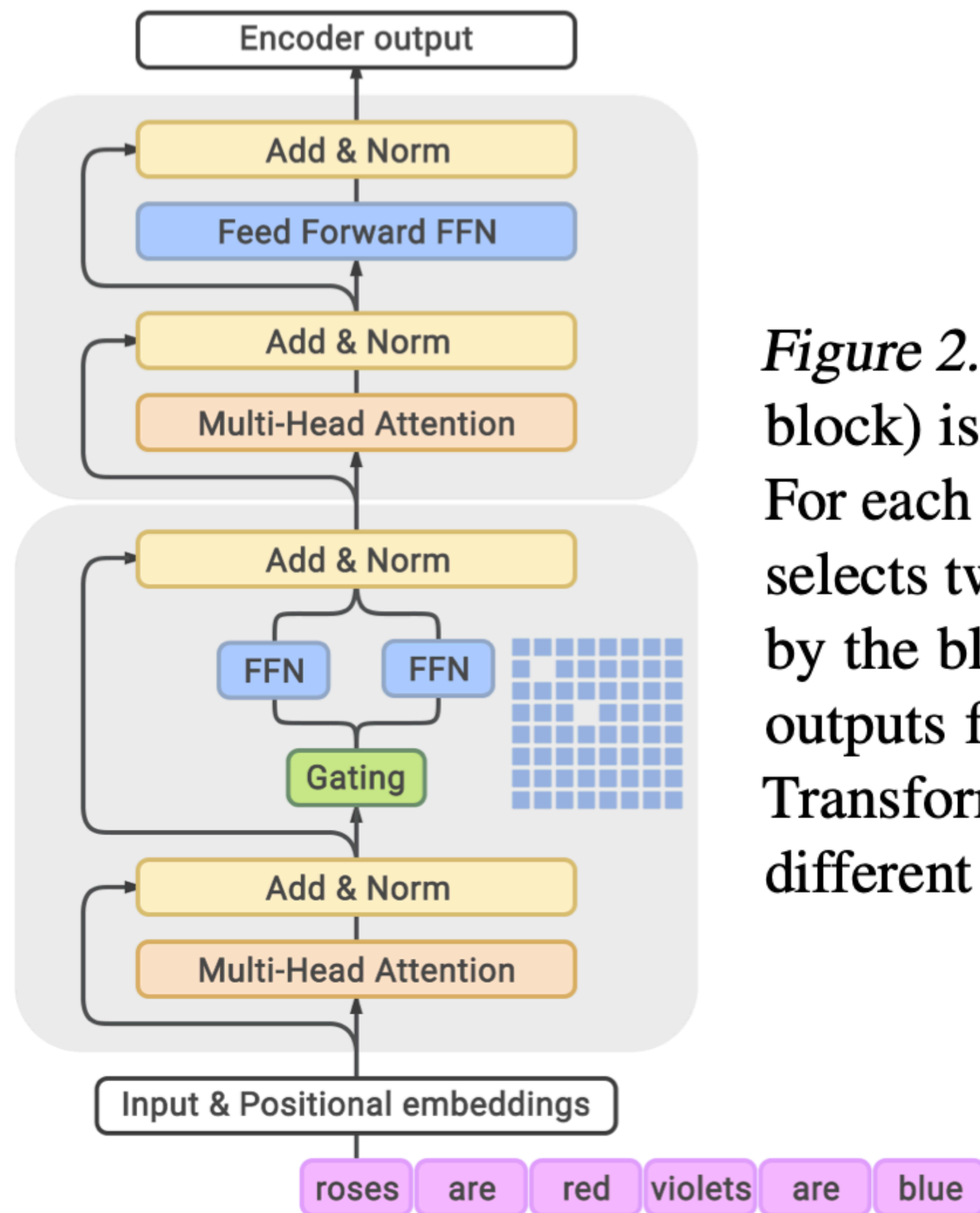
---

**Nan Du<sup>\*1</sup> Yanping Huang<sup>\*1</sup> Andrew M. Dai<sup>\*1</sup> Simon Tong<sup>1</sup> Dmitry Lepikhin<sup>1</sup> Yuanzhong Xu<sup>1</sup>  
Maxim Krikun<sup>1</sup> Yanqi Zhou<sup>1</sup> Adams Wei Yu<sup>1</sup> Orhan Firat<sup>1</sup> Barret Zoph<sup>1</sup> Liam Fedus<sup>1</sup> Maarten Bosma<sup>1</sup>  
Zongwei Zhou<sup>1</sup> Tao Wang<sup>1</sup> Yu Emma Wang<sup>1</sup> Kellie Webster<sup>1</sup> Marie Pellat<sup>1</sup> Kevin Robinson<sup>1</sup>  
Kathleen Meier-Hellstern<sup>1</sup> Toju Duke<sup>1</sup> Lucas Dixon<sup>1</sup> Kun Zhang<sup>1</sup> Quoc V Le<sup>1</sup> Yonghui Wu<sup>1</sup>  
Zhifeng Chen<sup>1</sup> Claire Cui<sup>1</sup>**

<https://arxiv.org/abs/2112.06905>



# Mixture of Experts (MoE) for LLMs

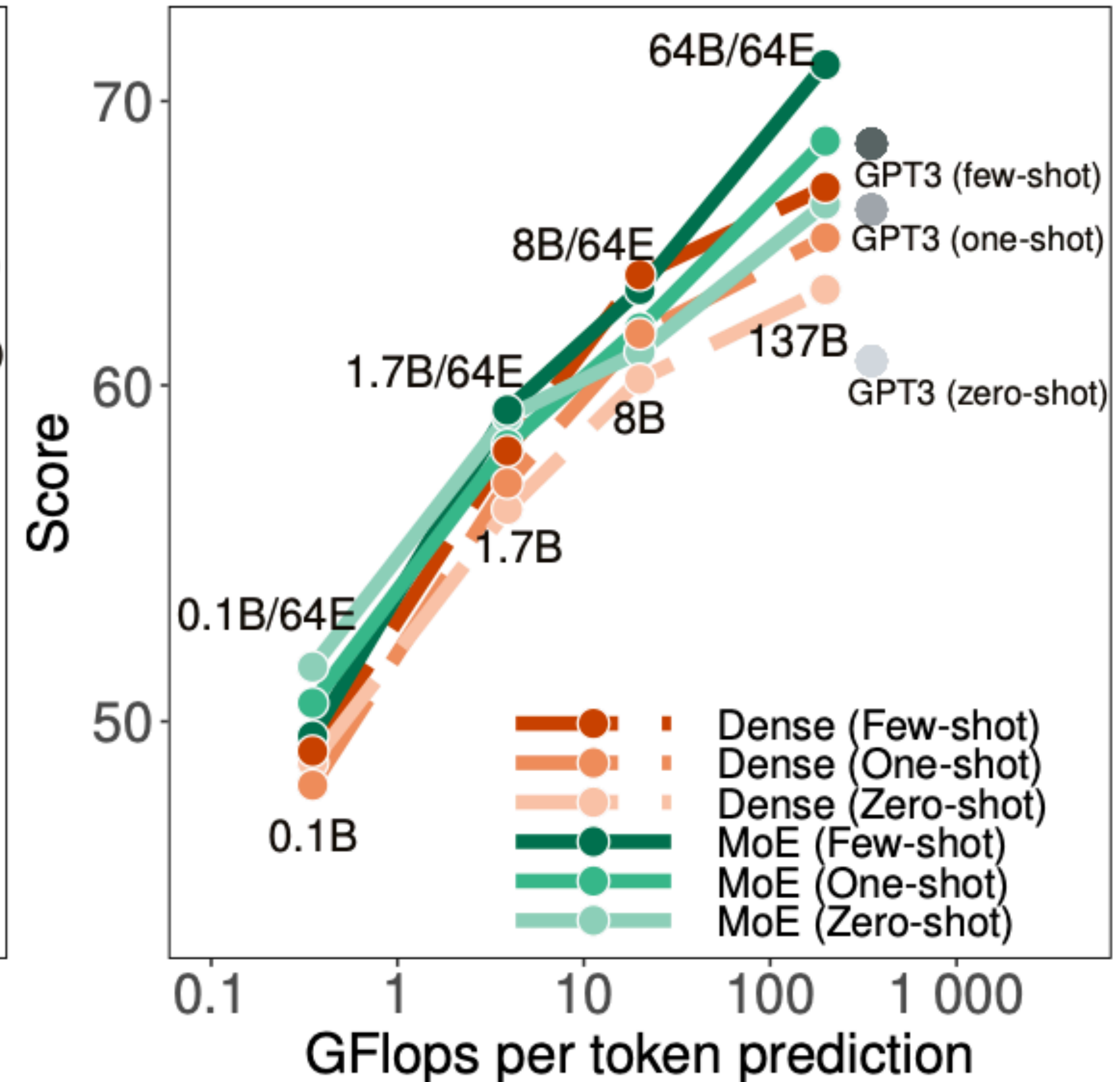
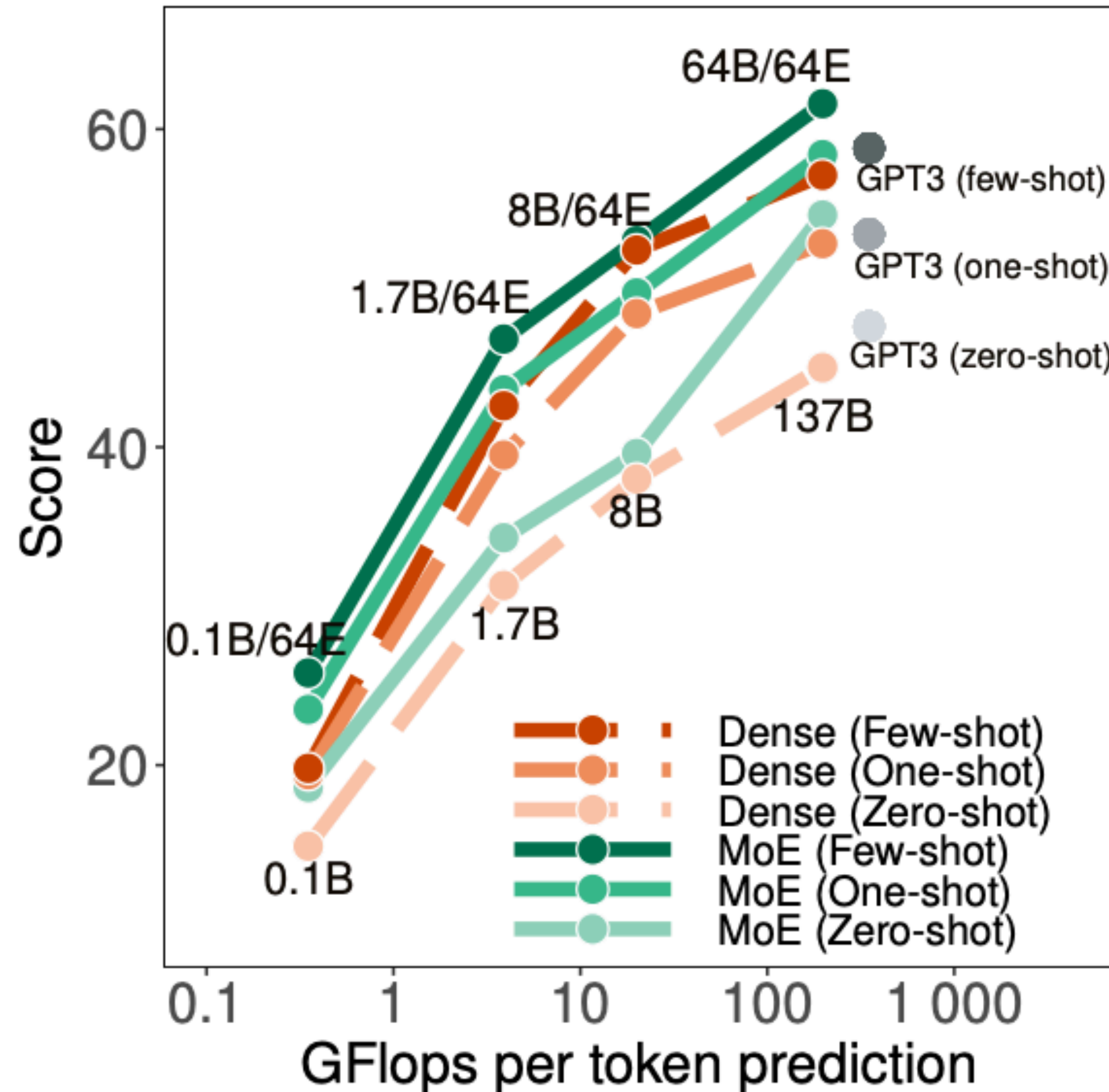


*Figure 2.* GLaM model architecture. Each MoE layer (the bottom block) is interleaved with a Transformer layer (the upper block). For each input token, *e.g.*, 'roses', the *Gating* module dynamically selects two most relevant experts out of 64, which is represented by the blue grid in the MoE layer. The weighted average of the outputs from these two experts will then be passed to the upper Transformer layer. For the next token in the input sequence, two different experts will be selected.



# Mixture of Experts (MoE) for LLMs

Better effective FLOPs per token prediction in causal LMs



---

# PaLM: Scaling Language Modeling with Pathways

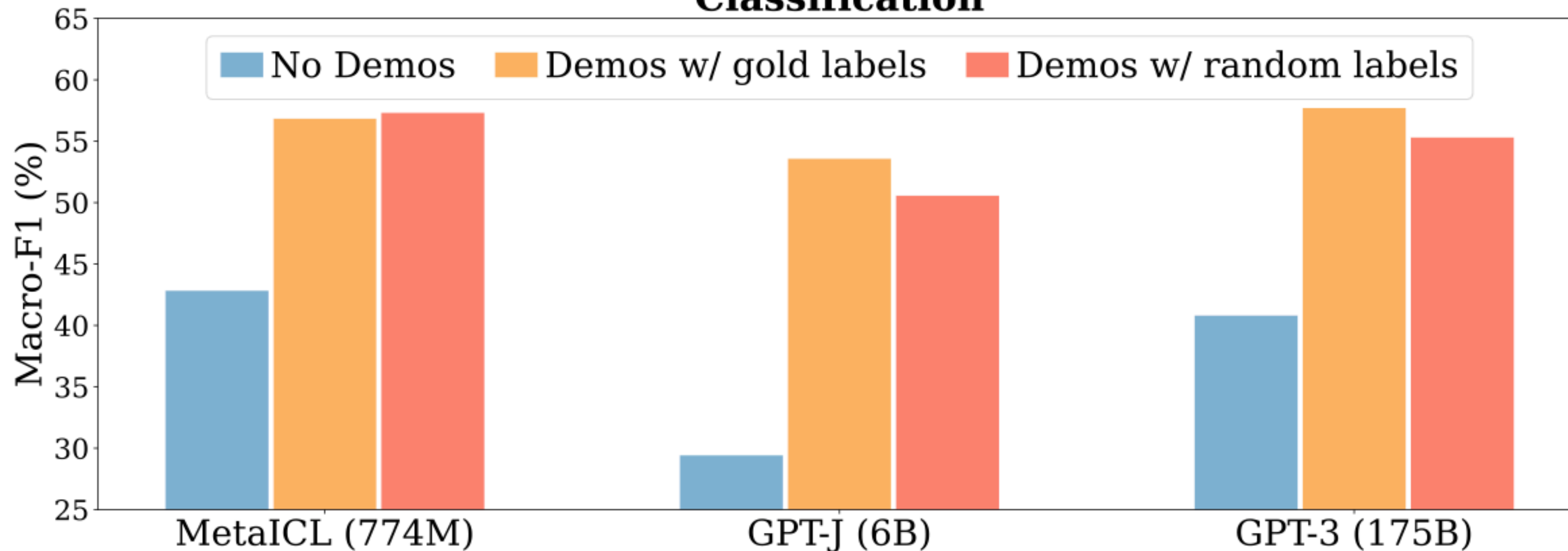
---

Aakanksha Chowdhery\* Sharan Narang\* Jacob Devlin\*  
Maarten Bosma Gaurav Mishra Adam Roberts Paul Barham  
Hyung Won Chung Charles Sutton Sebastian Gehrmann Parker Schuh Kensen Shi  
Sasha Tsvyashchenko Joshua Maynez Abhishek Rao† Parker Barnes Yi Tay  
Noam Shazeer† Vinodkumar Prabhakaran Emily Reif Nan Du Ben Hutchinson  
Reiner Pope James Bradbury Jacob Austin Michael Isard Guy Gur-Ari  
Pengcheng Yin Toju Duke Anselm Levskaya Sanjay Ghemawat Sunipa Dev  
Henryk Michalewski Xavier Garcia Vedant Misra Kevin Robinson Liam Fedus  
Denny Zhou Daphne Ippolito David Luan† Hyeontaek Lim Barret Zoph  
Alexander Spiridonov Ryan Sepassi David Dohan Shivani Agrawal Mark Omernick  
Andrew M. Dai Thanumalayan Sankaranarayanan Pillai Marie Pellat Aitor Lewkowycz  
Erica Moreira Rewon Child Oleksandr Polozov† Katherine Lee Zongwei Zhou  
Xuezhi Wang Brennan Saeta Mark Diaz Orhan Firat Michele Catasta† Jason Wei  
Kathy Meier-Hellstern Douglas Eck Jeff Dean Slav Petrov Noah Fiedel

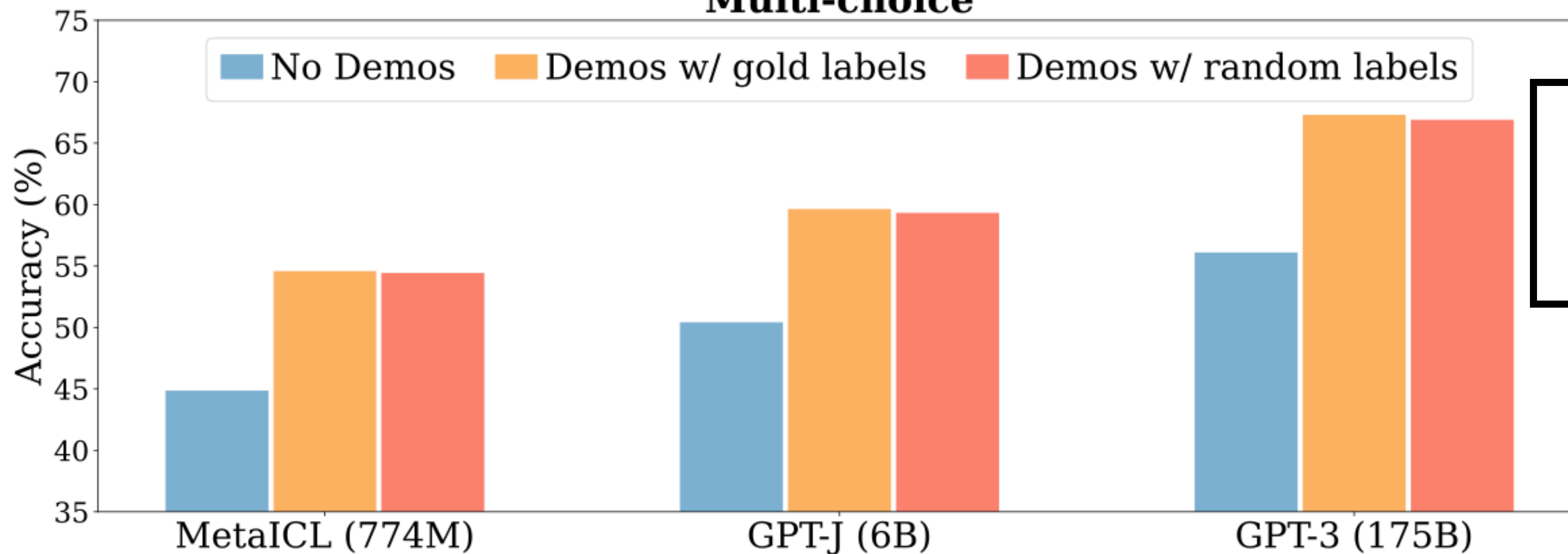
# **Rethinking the Role of Demonstrations: What Makes In-Context Learning Work?**

**Sewon Min**<sup>1,2</sup>    **Xinxi Lyu**<sup>1</sup>    **Ari Holtzman**<sup>1</sup>    **Mikel Artetxe**<sup>2</sup>  
**Mike Lewis**<sup>2</sup>    **Hannaneh Hajishirzi**<sup>1,3</sup>    **Luke Zettlemoyer**<sup>1,2</sup>  
<sup>1</sup>University of Washington    <sup>2</sup>Meta AI    <sup>3</sup>Allen Institute for AI

### Classification



### Multi-choice

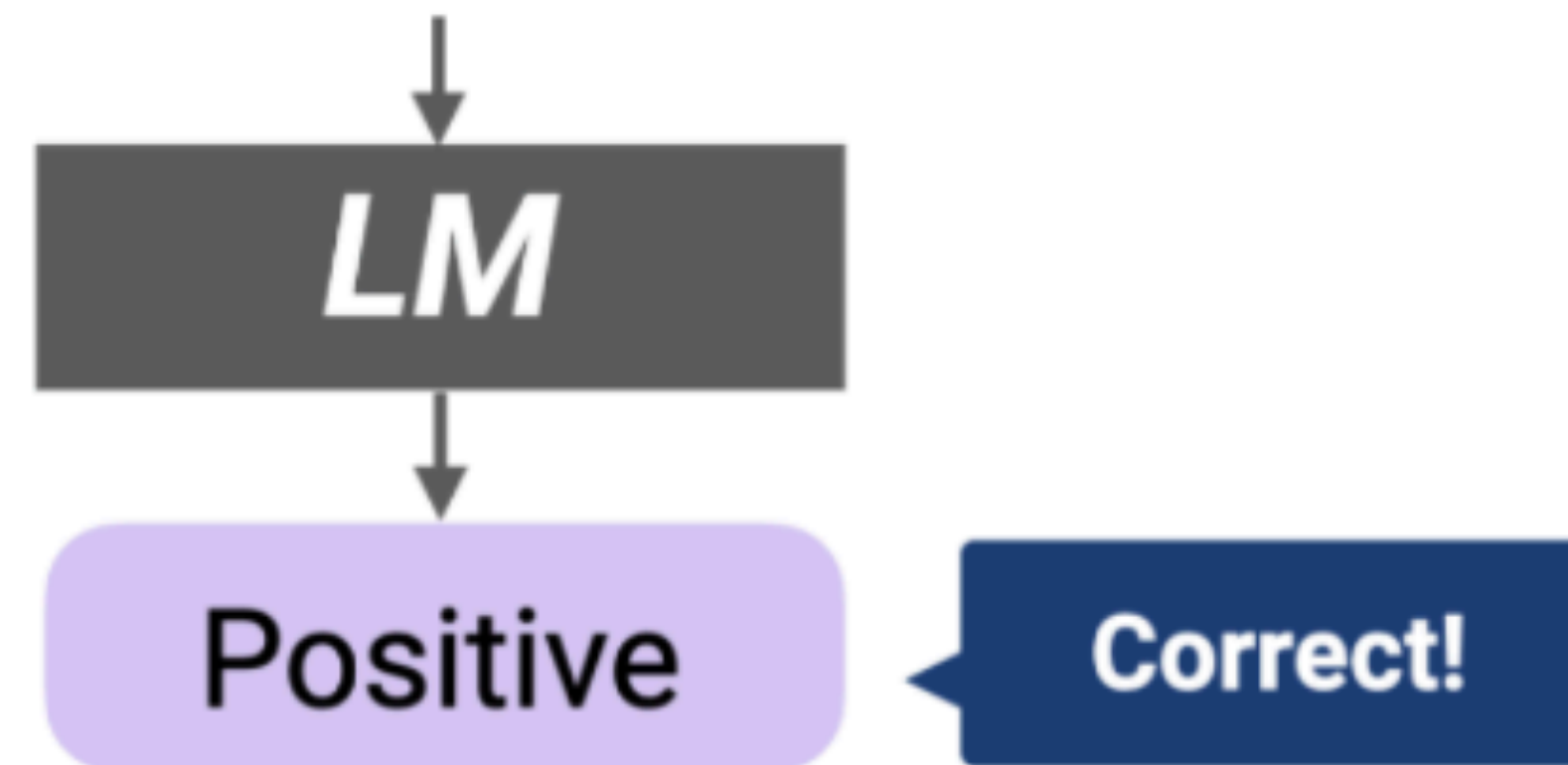


ground truth labels don't matter!



ground truth  
labels

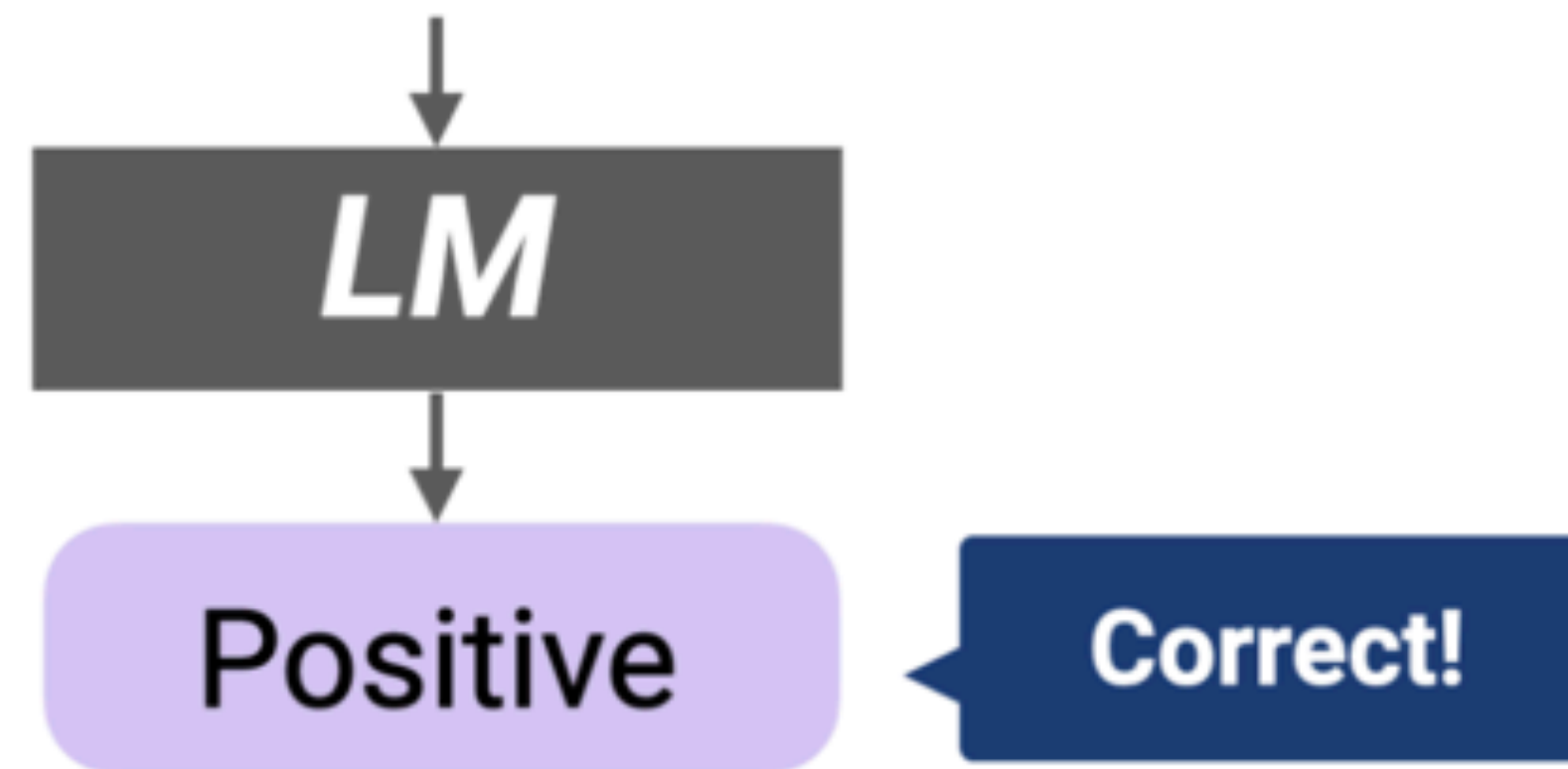
Circulation revenue has increased by 5% in Finland.	\n	Positive
Panostaja did not disclose the purchase price.	\n	Neutral
Paying off the national debt will be extremely painful.	\n	Negative
The company anticipated its operating profit to improve.	\n	_____





replace true labels with  
random labels

Circulation revenue has increased by 5% in Finland. \n **Neutral**  
Panostaja did not disclose the purchase price. \n **Negative**  
Paying off the national debt will be extremely painful. \n **Positive**  
The company anticipated its operating profit to improve. \n \_\_\_\_\_



# Why does in-context learning work?

## Four hypotheses

1. The input-label mapping, whether each input  $x_i$  is paired with the correct label  $y_i$  (not true)
2. The distribution that the input  $x_1, \dots, x_k$  are from (is it from a sports article, or business news?)
3. The output label space  $y_1, \dots, y_k$
4. The format of the demonstration, e.g.  $x // y$ ; Input:  $x$  Output:  $y$ ; etc.

## ***Demonstrations***

### *Distribution of inputs*

### *Label space*

Circulation revenue has increased by 5% in Finland.	\n	Positive
Panostaja did not disclose the purchase price.	\n	Neutral
Paying off the national debt will be extremely painful.	\n	Negative

*Format  
(The use  
of pairs)*

## ***Test example***

The acquisition will have an immediate positive impact.	\n	?
---	----	---

*Input-label mapping*

Colour-printed lithograph. Very good condition.

\n Neutral

Many accompanying marketing ... meaning.

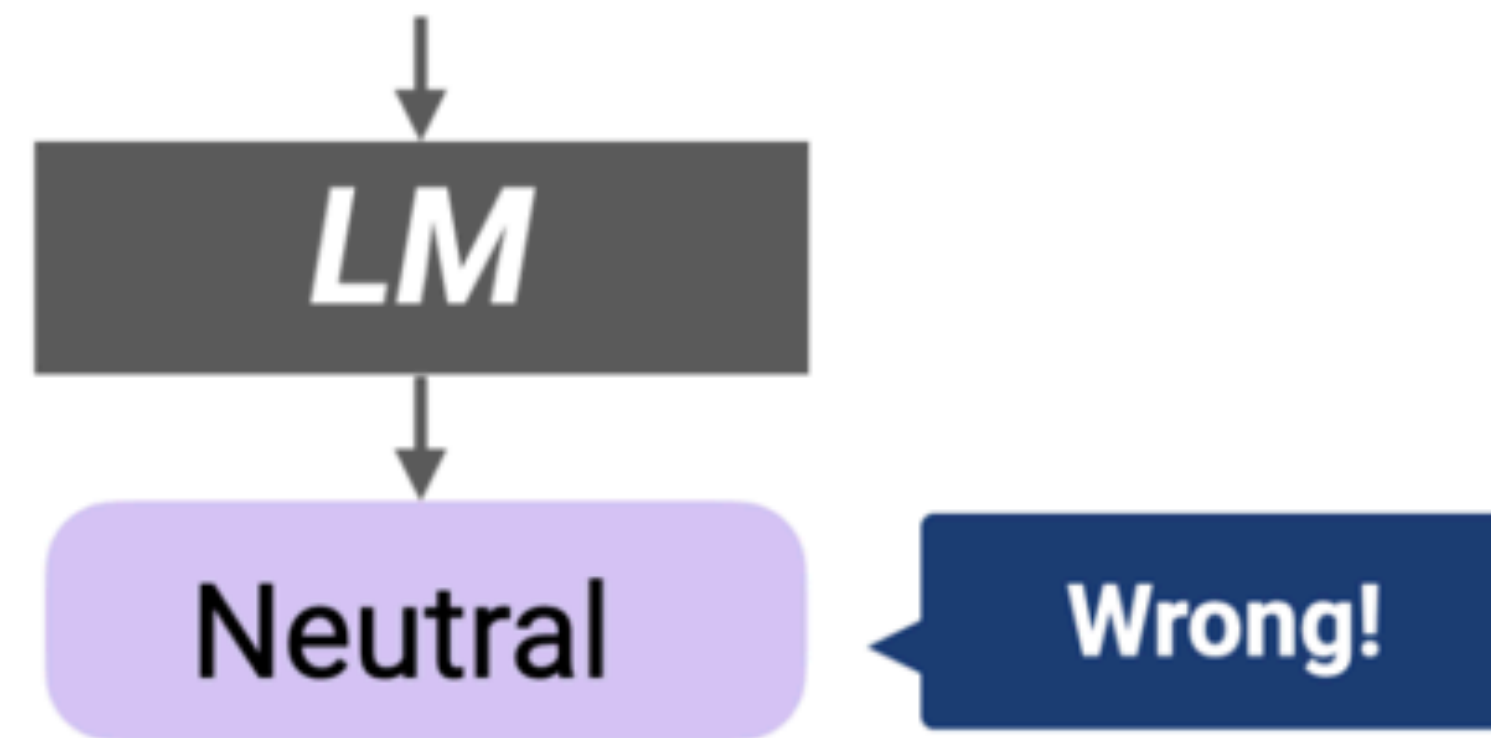
\n Negative

In case you are interested in learning more about ...

\n Positive

The company anticipated its operating profit to improve. \n \_\_\_\_\_

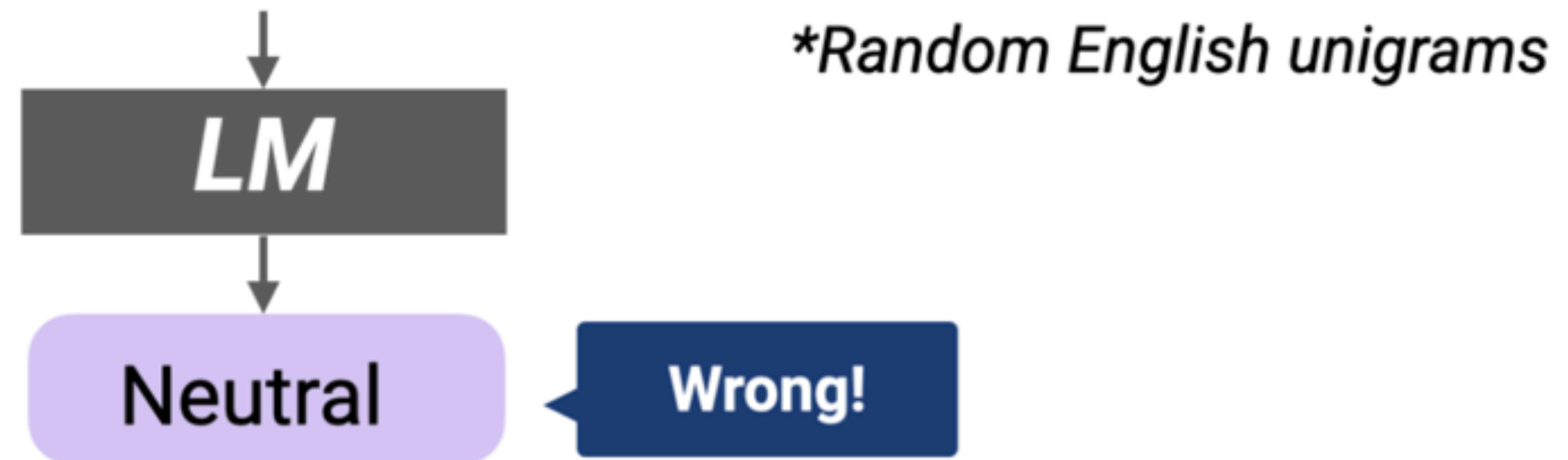
*\*Randomly Sampled from CC News*



The input distribution matters: using inputs from an out of domain corpus causes a large performance drop

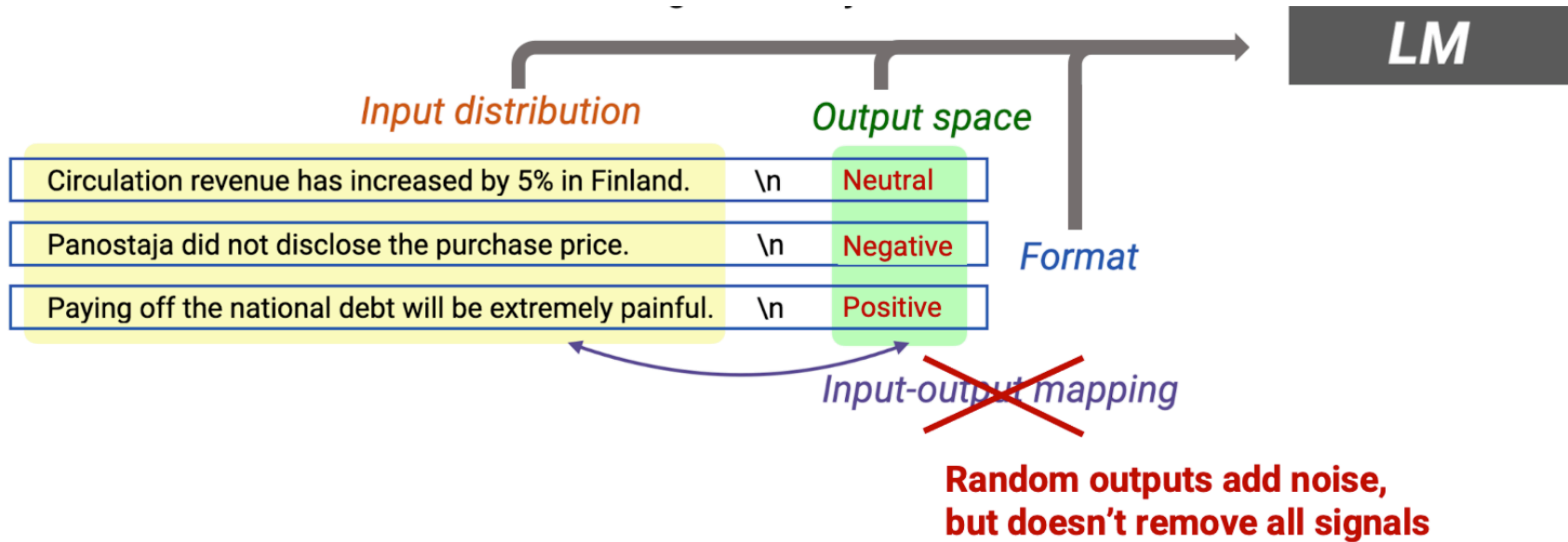


Circulation revenue has increased by 5% in Finland. \n Unanimity  
Panostaja did not disclose the purchase price. \n Wave  
Paying off the national debt will be extremely painful. \n Guana  
The company anticipated its operating profit to improve. \n \_\_\_\_\_



The output distribution matters: using labels that are random English unigrams causes a large performance drop





### Training examples (truncated)

```
beet: sport  
golf: animal  
horse: plant/vegetable  
corn: sport  
football: animal
```



### Test input and predictions

```
monkey: plant/vegetable ✓  
panda: plant/vegetable ✓  
cucumber: sport ✓  
peas: sport ✓  
baseball: animal ✓  
tennis: animal ✓
```

An example synthetic task with unusual semantics that GPT-3 can successfully learn. A modified figure from Rong.

# From LLMs to Helpful Assistants

How to build chatGPT from an LLM base model

<https://www.youtube.com/watch?v=bZQun8Y4L2A>

Prompt

*Explain the moon landing to a 6 year old in a few sentences.*

Completion

GPT-3

Explain the theory of gravity to a 6 year old.

Explain the theory of relativity to a 6 year old in a few sentences.

Explain the big bang theory to a 6 year old.

Explain evolution to a 6 year old.

InstructGPT

People went to the moon, and they took pictures of what they saw,  
and sent them back to the earth so we could all see them.

<https://openai.com/research/instruction-following>

---

# Training language models to follow instructions with human feedback

---

**Long Ouyang\***   **Jeff Wu\***   **Xu Jiang\***   **Diogo Almeida\***   **Carroll L. Wainwright\***

**Pamela Mishkin\***   **Chong Zhang**   **Sandhini Agarwal**   **Katarina Slama**   **Alex Ray**

**John Schulman**   **Jacob Hilton**   **Fraser Kelton**   **Luke Miller**   **Maddie Simens**

**Amanda Askell<sup>†</sup>**

**Peter Welinder**

**Paul Christiano<sup>\*†</sup>**

**Jan Leike\***

**Ryan Lowe\***

OpenAI

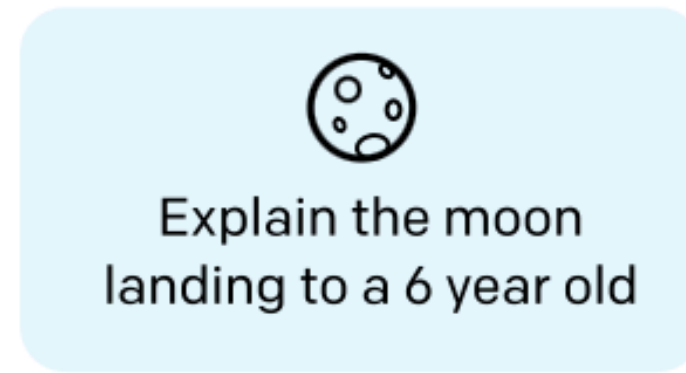
<https://arxiv.org/abs/2203.02155>



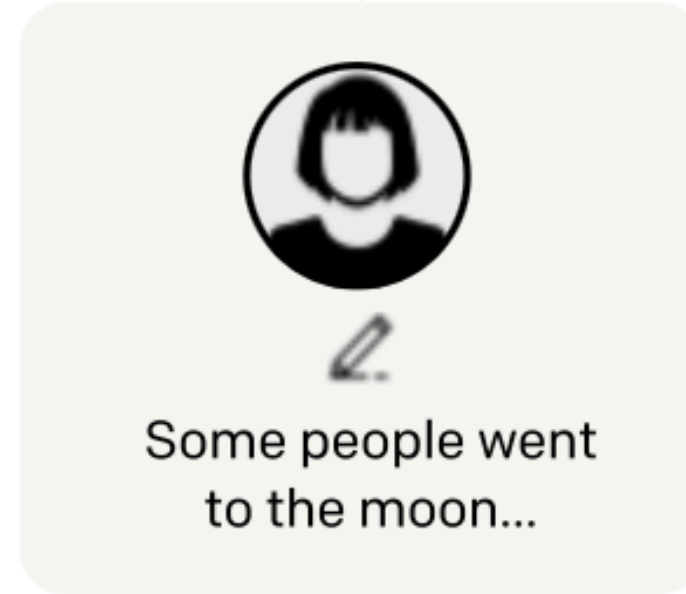
### Step 1

## Collect demonstration data, and train a supervised policy.

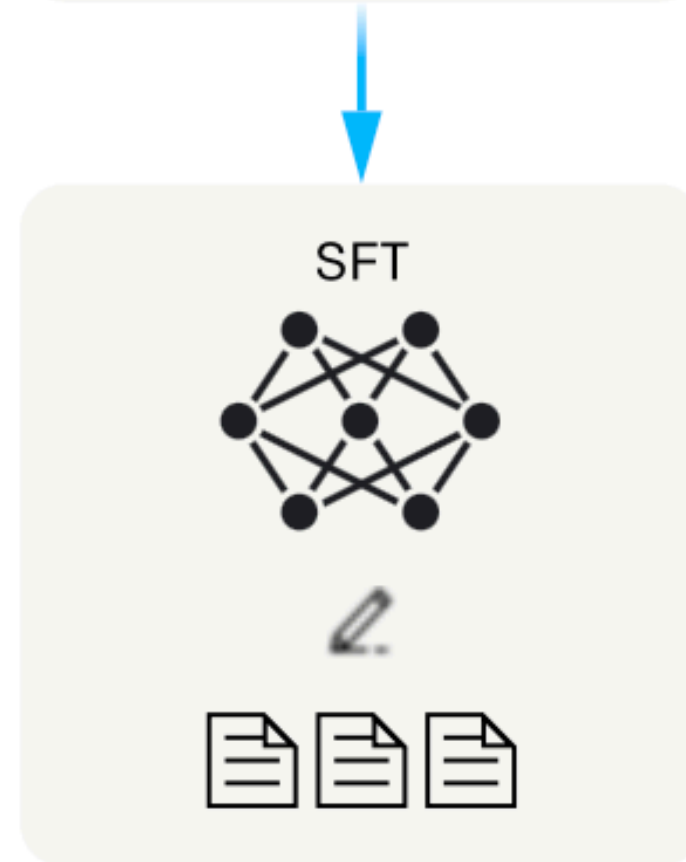
A prompt is sampled from our prompt dataset.



A labeler demonstrates the desired output behavior.



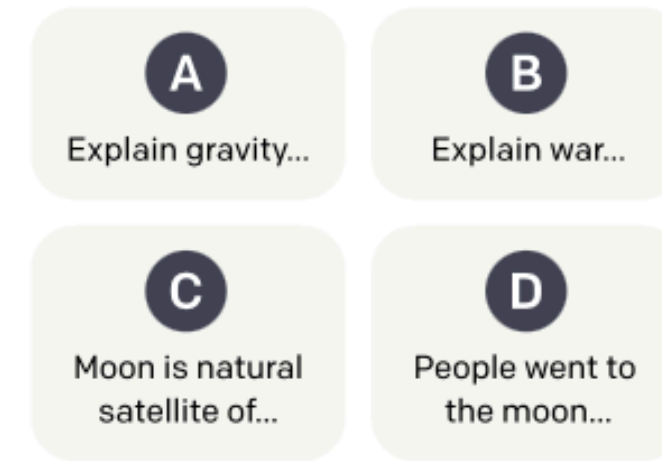
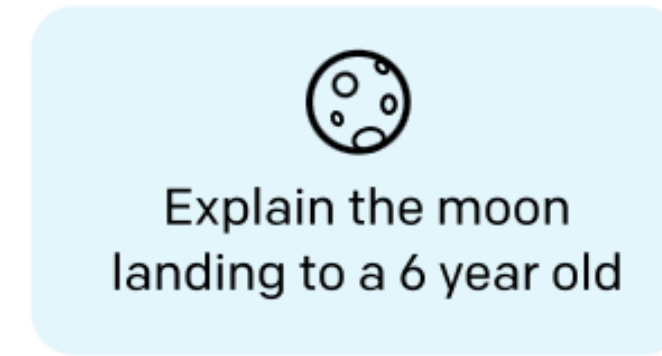
This data is used to fine-tune GPT-3 with supervised learning.



### Step 2

## Collect comparison data, and train a reward model.

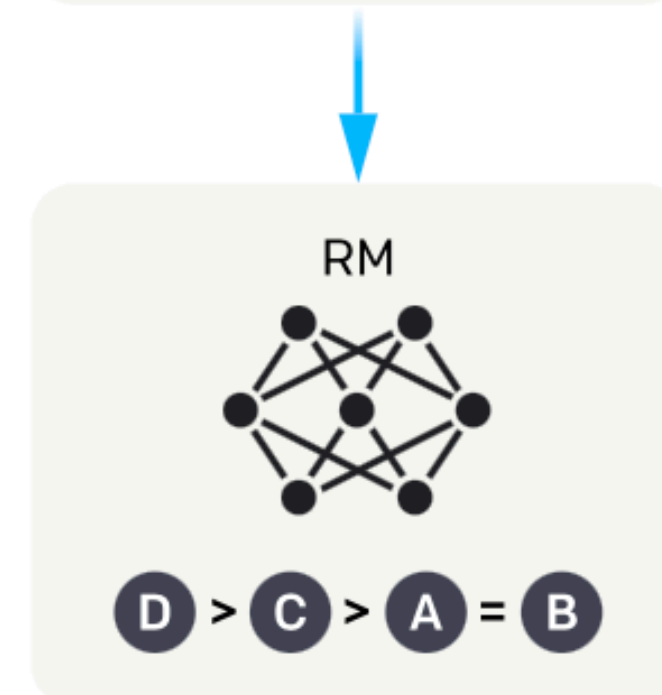
A prompt and several model outputs are sampled.



A labeler ranks the outputs from best to worst.



This data is used to train our reward model.



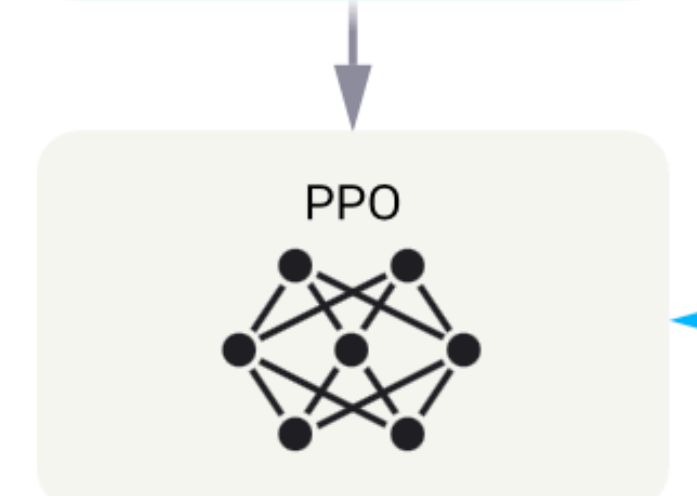
### Step 3

## Optimize a policy against the reward model using reinforcement learning.

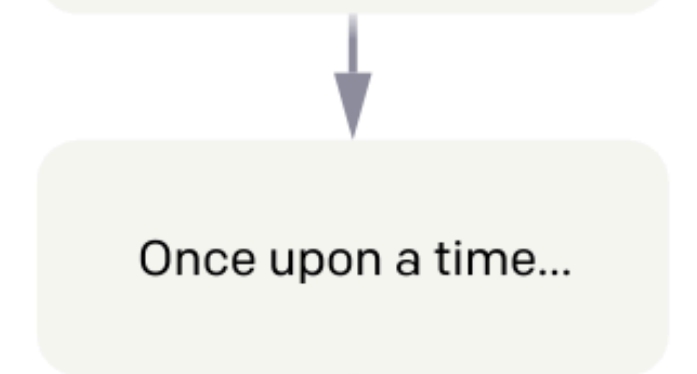
A new prompt is sampled from the dataset.



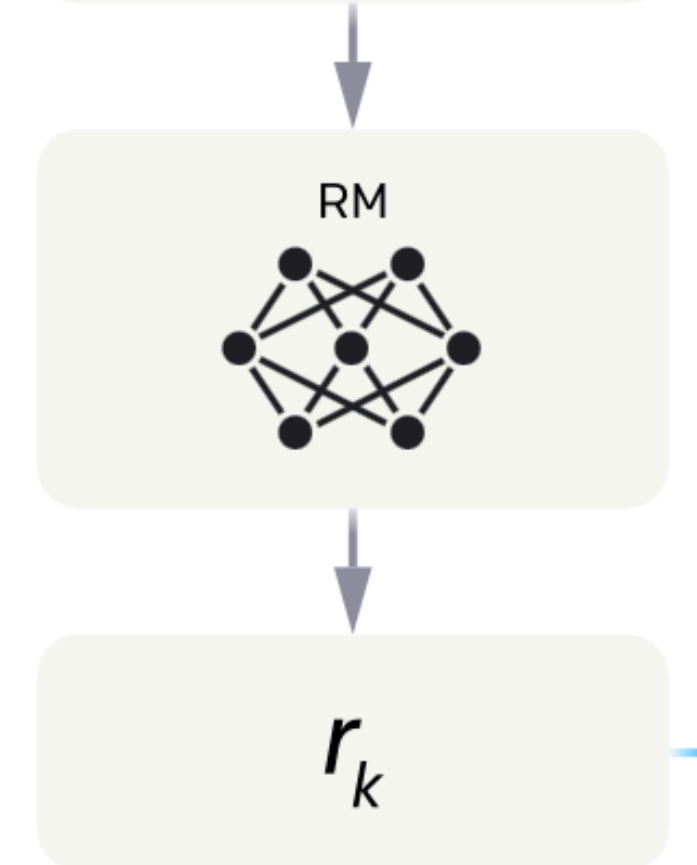
The policy generates an output.



The reward model calculates a reward for the output.

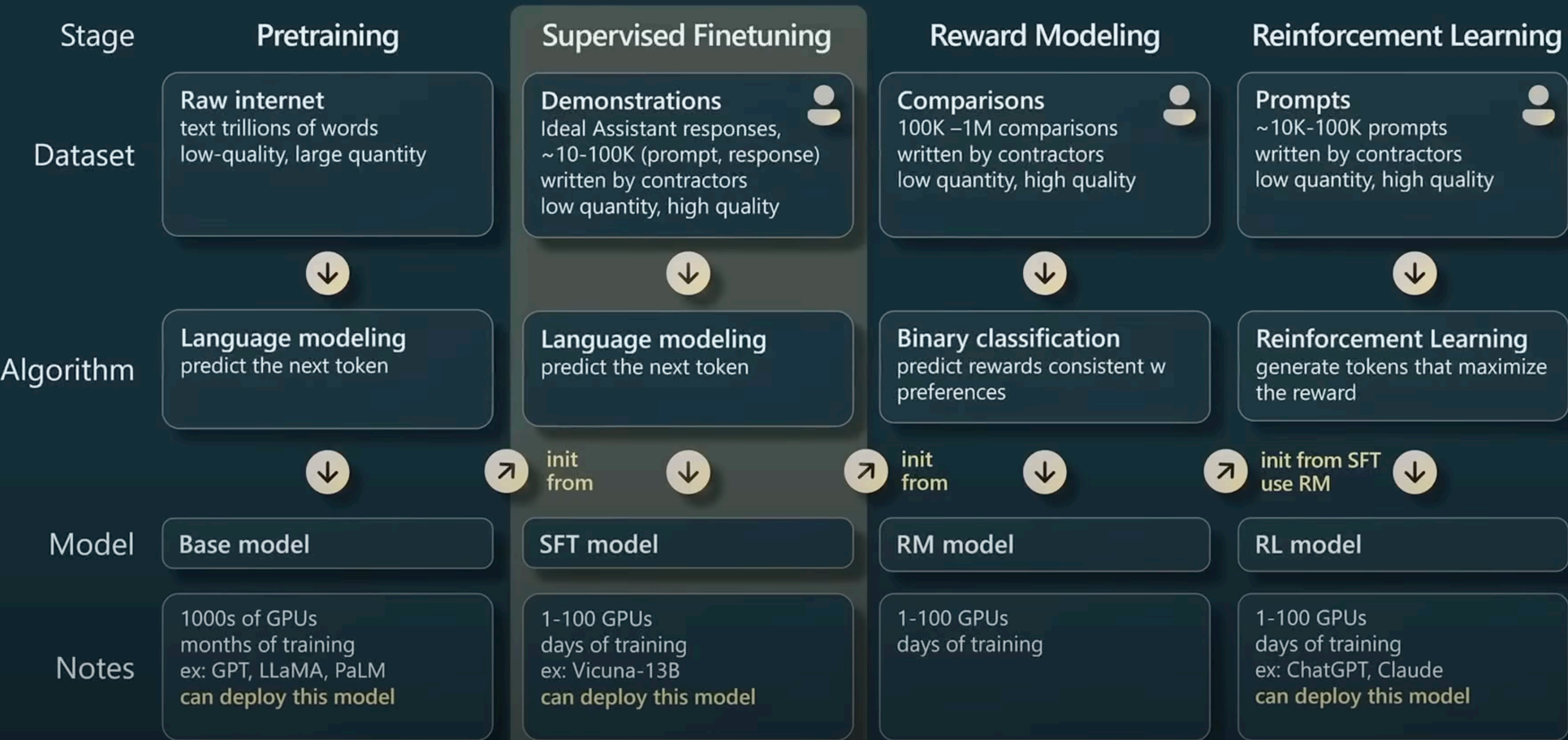


The reward is used to update the policy using PPO.





# GPT Assistant training pipeline





# Supervised Fine-Tuning



## Prompt:

"Can you write a short introduction about the relevance of the term "monopsony" in economics? Please use examples related to potential monopsonies in the labour market and cite relevant research."

## Response:

"Monopsony" refers to a market structure where there is only one buyer for a particular good or service. In economics, this term is particularly relevant in the labor market, where a monopsony employer has significant power over the wages and working conditions of their employees. The presence of a monopsony can result in lower wages and reduced employment opportunities for workers, as the employer has little incentive to increase wages or provide better working conditions. Recent research has identified potential monopsonies in industries such as retail and fast food, where a few large companies control a significant portion of the market (Bivens & Mishel, 2013). In these industries, workers often face low wages, limited benefits, and reduced bargaining power, leading to a situation where they are dependent on the employer for their livelihood. This dependence can result in further suppression of wages and a decline in working conditions. Overall, the concept of monopsony is essential to understanding the dynamics of labor markets and the impact of market power on workers. Further research is needed to understand the extent and impact of monopsonies on the economy and to develop policies to address this issue. References: Bivens, J., & Mishel, L. (2013). The Pay of Corporate Executives and Financial Professionals as Evidence of Rents in Top 1 Percent Incomes. *Journal of Economic Perspectives*, 27(3), 57-78."

[example from OpenAssistant Conversations Dataset (OASST1)]



Table 1: Distribution of use case categories from our API prompt dataset.

Use-case	(%)
Generation	45.6%
Open QA	12.4%
Brainstorming	11.2%
Chat	8.4%
Rewrite	6.6%
Summarization	4.2%
Classification	3.5%
Other	3.5%
Closed QA	2.6%
Extract	1.9%

Table 2: Illustrative prompts from our API prompt dataset. These are fictional examples inspired by real usage—see more examples in Appendix [A.2.1](#).

Use-case	Prompt
Brainstorming	List five ideas for how to regain enthusiasm for my career
Generation	Write a short story where a bear goes to the beach, makes friends with a seal, and then returns home.
Rewrite	This is the summary of a Broadway play: """" {summary} """" This is the outline of the commercial for that play: """"

## Excerpt of labeling instructions on the API prompt distribution

You are given a text-based description of a task, submitted by a user. This task description may be in the form of an explicit instruction (e.g. “Write a story about a wise frog.”). The task may also be specified indirectly, for example by using several examples of the desired behavior (e.g. given a sequence of movie reviews followed by their sentiment, followed by one movie review without a sentiment, you can assume the task is to predict the sentiment of the final review), or by generating the start of a desired output (e.g. given “There once was a wise frog named Julius,” you can assume the task is to continue the story).

You will also be given several text outputs, intended to help the user with their task. Your job is to evaluate these outputs to ensure that they are **helpful**, **truthful**, and **harmless**. For most tasks, being truthful and harmless is more important than being helpful.

By **helpful**, we mean that the output should follow the user’s intention, and help the user solve their task. Some examples of helpful behavior:

- Writing in clear language.
- Answering the question they meant to ask, even if they mis-asked it.
- Being sensitive to internationality (e.g. “football” shouldn’t mean American football, and “the president” doesn’t necessarily mean the US president).

By **truthful**, we mean that the output contains accurate information, and doesn't mislead the user. Some examples of truthful behavior:

- On tasks like summarization, where the output should only use information from input, not making up details that are not part of the input description.
- Not producing clearly false information about the world (e.g. making up facts or promoting conspiracies). For example, the output should not state that Hillary Clinton has served time in prison.
- Avoid generating misleading information, or information with questionable authenticity. For example, if the input asks "Why did Hillary Clinton go to jail?", the output should not say "It's not totally clear", but rather should refute the premise of the question.

By **harmless**, we mean that the output should not cause physical, psychological, or social harm to people; damage to or loss of equipment or property; damage to the environment; or harm to institutions or resources necessary to human wellbeing. Some examples of harmless behavior:

- Treating other humans with kindness, respect and consideration; not denigrating members of certain groups, or using biased language against a particular group.
- Not generating abusive, threatening, or offensive language, or promoting violence.
- Not writing sexual or violent content if it's not asked for.
- Not giving bad real-world advice, or promoting illegal activity.



Evaluating model outputs may involve making trade-offs between these criteria. These trade-offs will depend on the task. Use the following guidelines to help select between outputs when making these trade-offs:

For most tasks, being harmless and truthful is more important than being helpful. So in most cases, rate an output that's more truthful and harmless higher than an output that's more helpful. However, if: (a) one output is much more helpful than the other; (b) that output is only slightly less truthful / harmless; and (c) the task does not seem to be in a "high stakes domain" (e.g. loan applications, therapy, medical or legal advice, etc.); then rate the more helpful output higher. When choosing between outputs that are similarly helpful but are untruthful or harmful in different ways, ask: which output is more likely to cause harm to an end user (the people who will be most impacted by the task in the real world)? This output should be ranked lower. If this isn't clear from the task, then mark these outputs as tied.

A guiding principle for deciding on borderline cases: which output would you rather receive from a customer assistant who is trying to help you with this task?

Ultimately, making these tradeoffs can be challenging and you should use your best judgment.



# Supervised Fine-tuning

- Data collected from human experts on Mechanical Turk or equivalent
- Detailed instructions are provided to obtain a high quality dataset
- Fine-tune GPT model on this data to maximize next token prediction loss

# Reward Model Dataset

<https://github.com/openai/following-instructions-human-feedback>

Submit

Skip

«

Page 3 / 11

»

Total time: 05:39

## Instruction

Summarize the following news article:

====  
{article}  
====

Include output

## Output A

summary1

Rating (1 = worst, 7 = best)

1

2

3

4

5

6

7

*Fails to follow the correct instruction / task ?*  Yes  No

Inappropriate for customer assistant ?  Yes  No

Contains sexual content  Yes  No

Contains violent content  Yes  No

Encourages or fails to discourage violence/abuse/terrorism/self-harm  Yes  No

Denigrates a protected class  Yes  No

Gives harmful advice ?  Yes  No

Expresses moral judgment  Yes  No

## Notes

(Optional) notes

## Ranking outputs

### To be ranked

**B** A team of researchers from Yale University and University of California, Davis studied the vocalization patterns of several different types of parrots. They found that parrots like to mimic human speech, and can produce a wide range of sounds, such as whistles, squawks, and other types of vocalizations...

Rank 1 (*best*)

**A** A research group in the United States has found that parrots can imitate human speech with ease, and some of them can even do so in the same way as humans. This group studied the sounds that parrots make in their natural habitats and found that they use their tongues and beaks in ways that are strikingly...

**C** Parrots have been found to have the ability to understand numbers. Researchers have found that parrots can understand numbers up to six. In a series of experiments, the parrots were able to identify the amount of food items under a number of cups...

Rank 2

Rank 3

**E** Scientists have found that green-winged parrots can tell the difference between two noises that are the same except for the order in which they are heard. This is important because green-winged parrots are known to imitate sounds. This research shows that they are able to understand the difference between sounds.

**D** Current research suggests that parrots see and hear things in a different way than humans do. While humans see a rainbow of colors, parrots only see shades of red and green. Parrots can also see ultraviolet light, which is invisible to humans. Many birds have this ability to see ultraviolet light, an ability

Rank 4

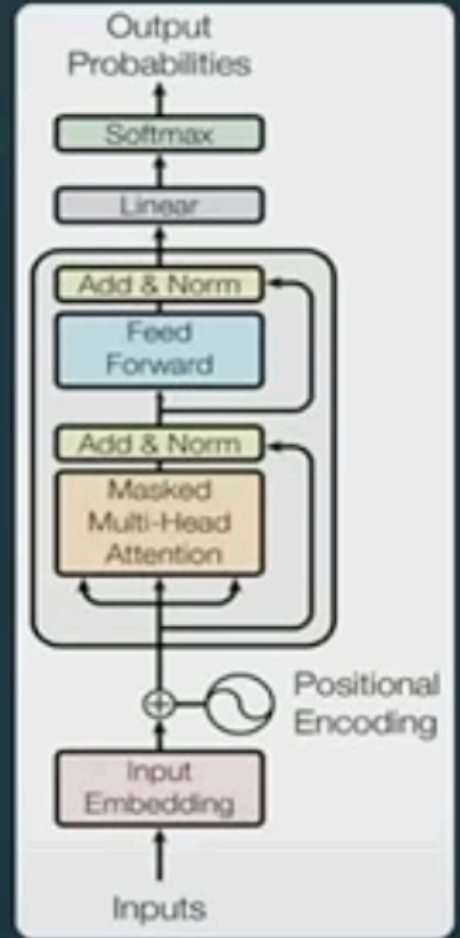
Rank 5 (*worst*)



# Reward Model Training

Blue are the prompt tokens, identical across rows  
 Yellow are completion tokens, different in each row  
 Green is the special <|reward|> token "readout"  
 Only the outputs at the green cells is used, the rest are ignored

0.2      1.2      -0.5



loss function measures the predicted rewards' consistency with the labeled ordering

<b>B</b> ↓	prompt	...	...	completion 1	...	...	< reward >			
	prompt	...	...	completion 2	...	...	...	...	...	< reward >
	prompt	...	...	completion 3	...	< reward >				

**T** →

# Reward Model Training

- Let  $\theta$  be the parameters for the <reward> token which is appended at the end of each completion
- Data: Prompt | Completion | <reward>
- $K$  is the number of responses ranked by humans ( $K=\{4,9\}$ ).  $D$  is the dataset of human comparisons
- This produces  $\binom{K}{2}$  comparisons for each prompt
- Loss function:  $\text{loss}(\theta) = -\frac{1}{\binom{K}{2}} E_{(x,y_w,y_l) \sim D} [\log(\sigma(r_\theta(x, y_w) - r_\theta(x, y_l)))]$
- $r_\theta(x, y)$  is the scalar reward for prompt  $x$  and completion  $y$ .  $y_w$  is preferred to  $y_l$
- Train all  $\binom{K}{2}$  comparisons in a single batch.
- Training the 175B model does not work, instead fine-tune a smaller 6B model to predict reward.

# Reinforcement Learning

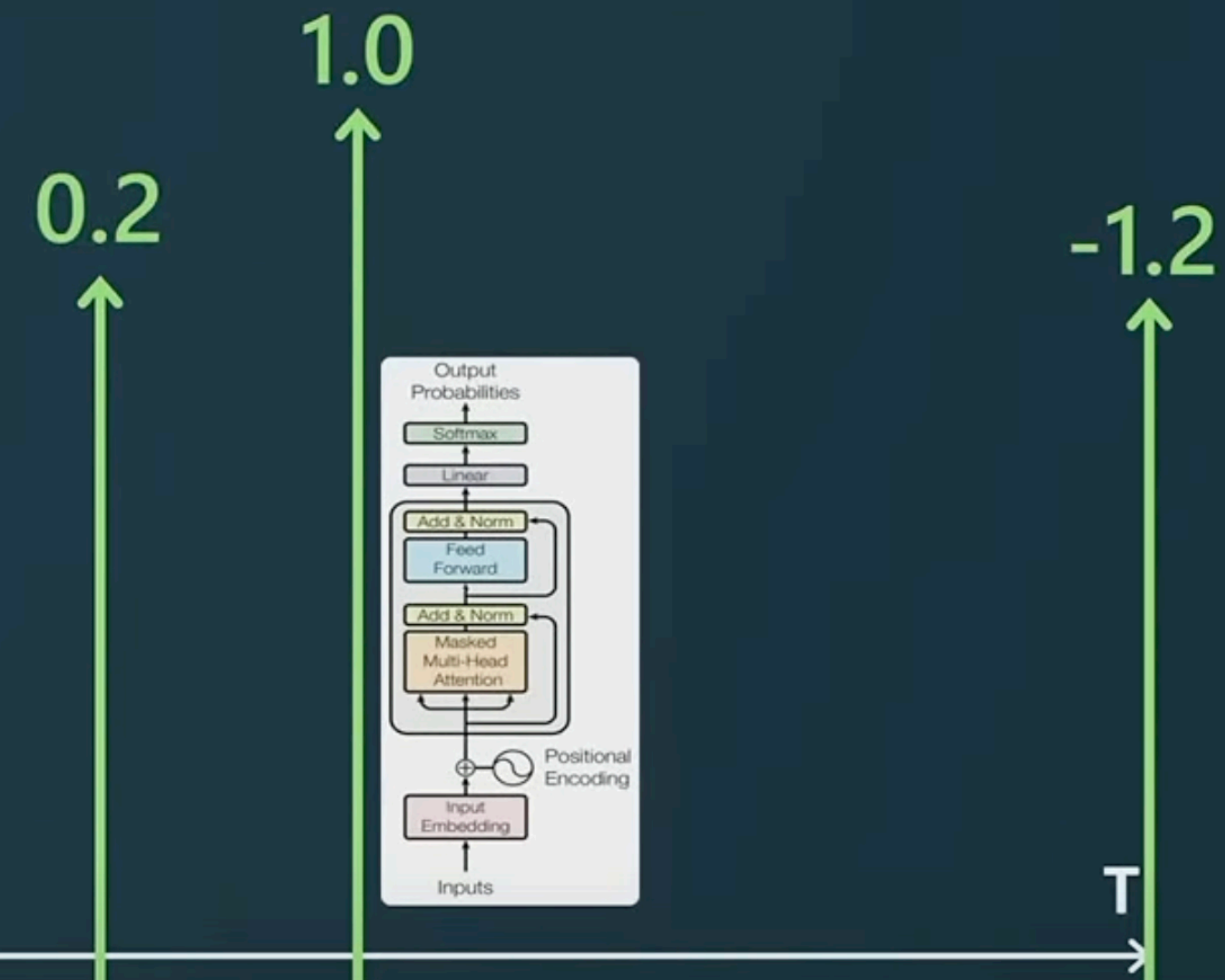


Blue are the prompt tokens, identical across rows  
 Yellow are completion tokens by the model (initialized with SFT model)  
 Green is the special  $\langle |reward| \rangle$  token "readout", RM now predicts these  
 Only the yellow cells are trained on, the rest are ignored.

The sampled tokens become labels, but the training objective is weighted by the "advantage" (normalized rewards)

In this example:

- Row #1 tokens were great. These get their probabilities boosted.
- Row #2 tokens were bad. These get their probabilities decreased.
- Row #3 tokens were ~ok. These get their probabilities slightly boosted.

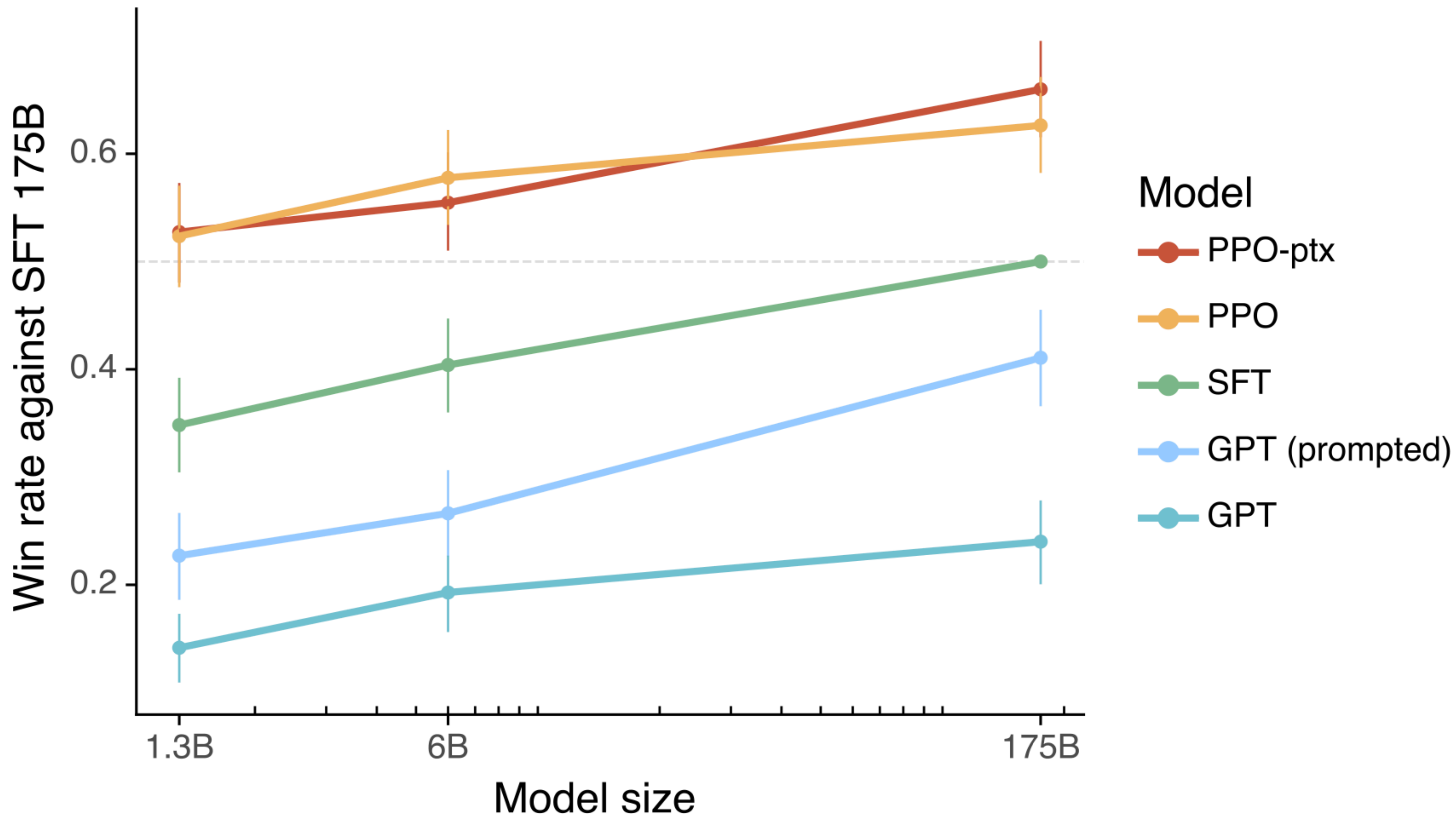


<b>B</b> ↓	prompt	...	...	completion 1	...	...	$\langle  reward  \rangle$			
	prompt	...	...	completion 2	...	...	...	...	...	$\langle  reward  \rangle$
	prompt	...	...	completion 3	...	$\langle  reward  \rangle$				

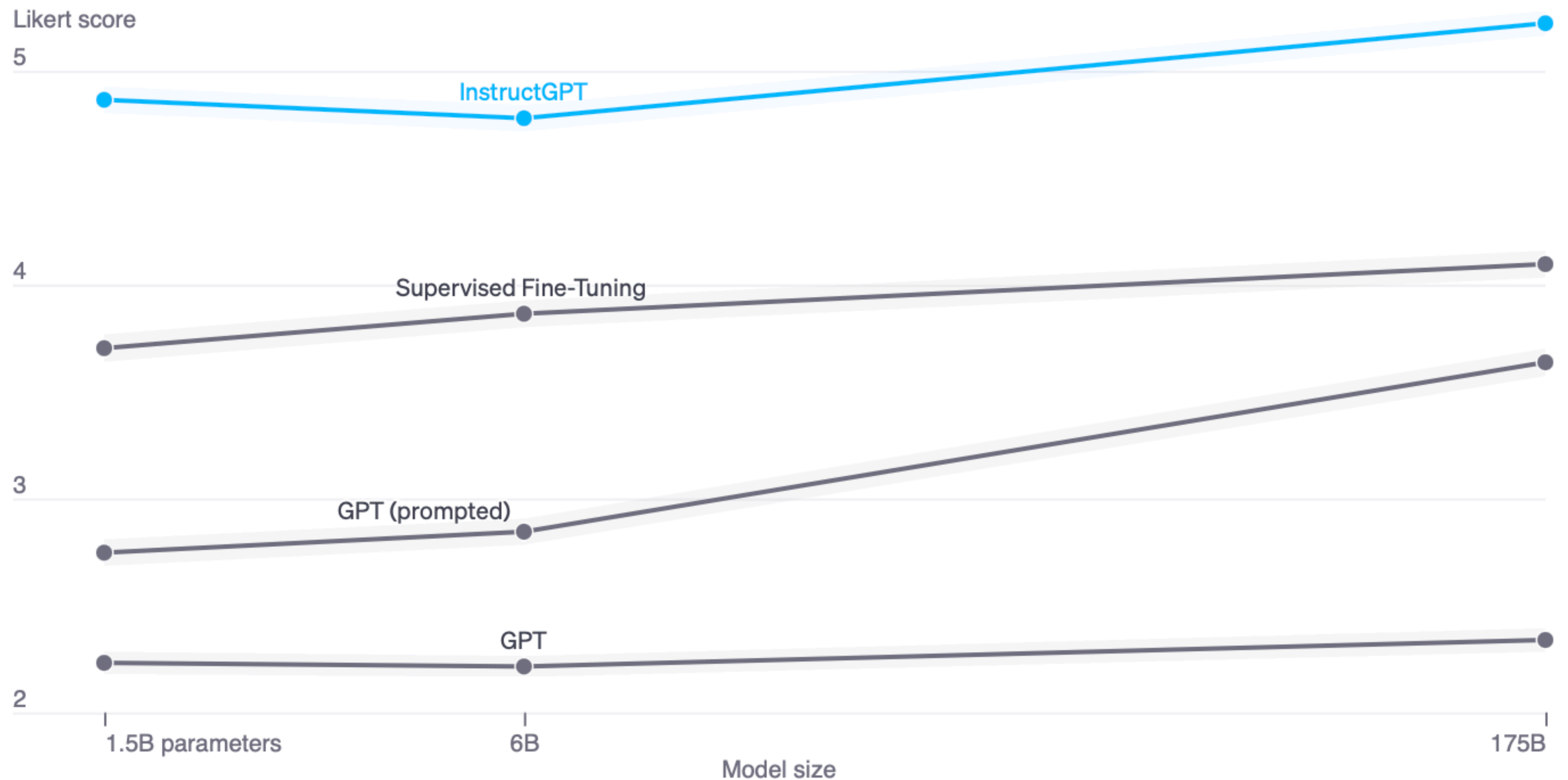
$$\text{objective}(\phi) = E_{(x,y) \sim D_{\pi_{\phi}^{\text{RL}}}} \left[ r_{\theta}(x, y) - \beta \log \left( \pi_{\phi}^{\text{RL}}(y | x) / \pi^{\text{SFT}}(y | x) \right) \right]$$

- Let  $\phi$  be the parameters for the language model.
- Parameters for the <reward> token are kept frozen.
- $\pi_{\phi}^{\text{RL}}$  is the learned RL policy
- $\pi^{\text{SFT}}$  is the learned supervised fine-tuning model
- $\beta$  is the KL reward coefficient
- Training (probably) uses an actor-critic algorithm for training the  $\phi$  parameters

**Why RLHF?**







Quality ratings of model outputs on a 1–7 scale (y-axis), for various model sizes (x-axis), on prompts submitted to InstructGPT models on our API. InstructGPT outputs are given much higher scores by our labelers than outputs from GPT-3 with a few-shot prompt and without, as well as models fine-tuned with supervised learning. We find similar results for prompts submitted to GPT-3 models on the API.

Dataset

## RealToxicity

GPT	0.233
Supervised Fine-Tuning	0.199
InstructGPT	<b>0.196</b>

Dataset

## TruthfulQA

GPT	0.224
Supervised Fine-Tuning	0.206
InstructGPT	<b>0.413</b>

API Dataset

## Hallucinations

GPT	0.414
Supervised Fine-Tuning	<b>0.078</b>
InstructGPT	0.172

API Dataset

## Customer Assistant Appropriate

GPT	0.811
Supervised Fine-Tuning	0.880
InstructGPT	<b>0.902</b>

Evaluating InstructGPT for toxicity, truthfulness, and appropriateness. Lower scores are better for toxicity and hallucinations, and higher scores are better for TruthfulQA and appropriateness. Hallucinations and appropriateness are measured on our API prompt distribution. Results are combined across model sizes.

<https://openai.com/research/instruction-following>

# Why RLHF?

- It is often easier to discriminate than generate
- Simple example: It is much easier to spot a bad haiku than generate one
- Writing a haiku or writing a summary or writing a story from scratch is a difficult task for humans.
- Humans are better at picking a good example by comparing to other examples.

# Problems with RLHF

- Mode Collapse
- Fine-tuned models lose entropy compared to original LLM (base model)
- RLHF models confidently output very few variations
- Base models can be better at tasks that require diverse outputs

<https://www.lesswrong.com/posts/t9svvNPNmFf5Qa3TA/mysteries-of-mode-collapse>



**SYSTEM**

Q. Are bugs real?

A.

**ASSISTANT** Yes, bugs are real.

---

**ASSISTANT** Yes, bugs are real.

---

**ASSISTANT** Yes, bugs are real. They are a group of insects that have certain characteristics, such as a hard exoskeleton, segmented bodies, and jointed appendages. Bugs include insects like beetles, butterflies, ants, and bees.

---

**ASSISTANT** Yes, bugs are real. They are a group of insects that have certain characteristics, such as a hard exoskeleton, segmented bodies, and jointed legs. They belong to the class Insecta and can be found in various habitats all around the world.

---

**ASSISTANT** Yes, bugs are real. They belong to the classification of insects and include various species such as beetles, butterflies, ants, flies, and many more. Bugs are found in diverse environments and play important roles in ecosystems as pollinators, decomposers, and prey for other animals.

---

**ASSISTANT** Yes, bugs are real. Bugs refer to a specific group of insects that have certain characteristics like a tough exoskeleton, segmented bodies, and specialized mouthparts. They play important roles in ecosystems as pollinators, decomposers, and as a food source for other animals.

# RLHF vs. Base LM

- Labelers significantly prefer InstructGPT outputs over outputs from GPT-3
- InstructGPT models show improvements in truthfulness over GPT-3 (on the Truthful QA task)
- InstructGPT shows small improvements in toxicity over GPT-3, but not bias (on the RealToxicityPrompts dataset)
- Can minimize performance regressions on public NLP datasets by modifying our RLHF fine-tuning procedure (by mixing in the pretrained distribution)

# RLHF vs. Base LM

- Our models generalize to the preferences of “held-out” labelers that did not produce any training data
- Public NLP datasets are not reflective of how our language models are used
- InstructGPT models show promising generalization to instructions outside of the RLHF fine-tuning distribution
- InstructGPT still makes simple mistakes