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

https://openai.com/research/language-unsupervised **Jun 2018**

GPT1

Pre-training an autoregressive language model

BooksCorpus: 7K unpublished books (1B words)

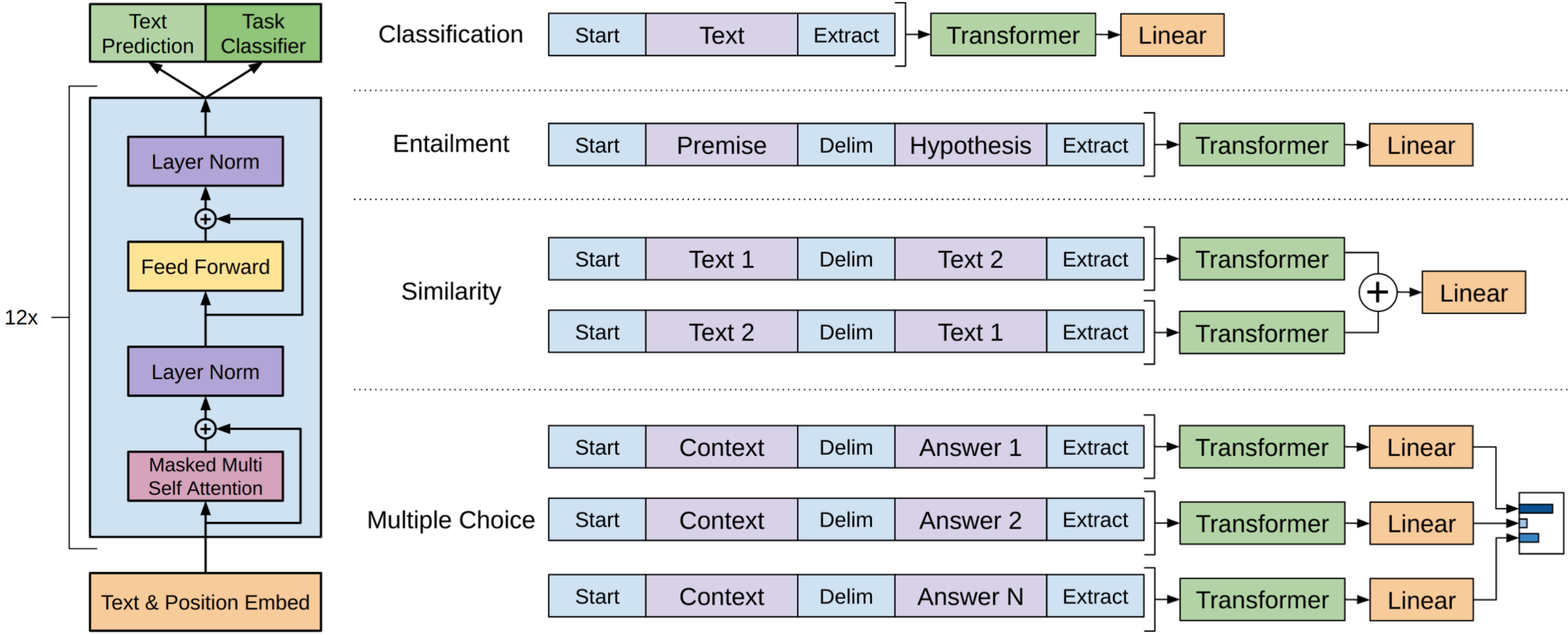
- Start with a large amount of unlabeled data $\mathcal{U} = \{u_1, \ldots, u_n\}$
- Pre-training objective: Maximize the likelihood of predicting the next token
  - $L_i(\mathcal{U}) = \sum_i \log P(u_i \mid u_{i-k}, \ldots, u_{i-1}; \Theta)$
- This is equivalent to training a Transformer decoder
  - $h_0 = UW_e + W_p$
  - $h_\ell = \text{transformer\_block}(h_{\ell-1}) \forall \ell \in [1,n]$
  - $P(u) = \text{softmax}(h_n W_e^T)$
- Directionality is needed to generate a well-formed probability distribution

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

n is the number of Transformer layers

W_e is the token embedding matrix

W_p is the position embedding matrix



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

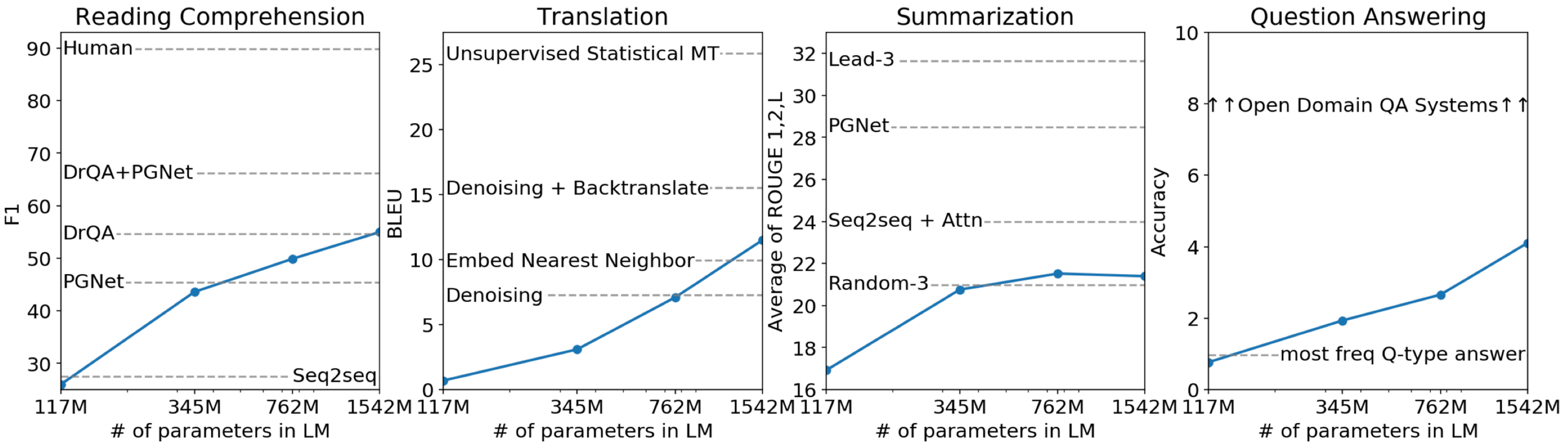
The GPT2 paper

Language Models are Unsupervised Multitask Learners

Alec Radford [*1] **Jeffrey Wu** [*1] **Rewon Child** [1] **David Luan** [1] **Dario Amodei** [**1] **Ilya Sutskever** [**1]

https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf

Feb 2019


Reading Comprehension
F1
90
80
70
60
50
40
30
Human
DrQA+PGNet
DrQA
PGNet
Seq2seq
117M
345M
762M
1542M
# of parameters in LM
Translation
BLEU
25
20
15
10
5
0
Unsupervised Statistical MT
Denoising + Backtranslate
Embed Nearest Neighbor
Denoising
117M
345M
762M
1542M
# of parameters in LM
Summarization
Average of ROUGE 1,2,L
32
30
28
26
24
22
20
18
16
Lead-3
PGNet
Seq2seq + Attn
Random-3
117M
345M
762M
1542M
# of parameters in LM
Question Answering
Accuracy
10
8
6
4
2
0
↑ ↑ Open Domain QA Systems ↑ ↑
most freq Q-type answer
117M
345M
762M
1542M
# of parameters in LM

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.

Results using Fine-tuning

	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	**21.8**
117M	**35.13**	45.99	**87.65**	**83.4**	**29.41**	65.85	1.16	1.17	37.50	75.20
345M	**15.60**	55.48	**92.35**	**87.1**	**22.76**	47.33	1.01	**1.06**	26.37	55.72
762M	**10.87**	**60.12**	**93.45**	**88.0**	**19.93**	**40.31**	**0.97**	**1.02**	22.05	44.575
1542M	**8.63**	**63.24**	**93.30**	**89.05**	**18.34**	**35.76**	**0.93**	**0.98**	**17.48**	42.16

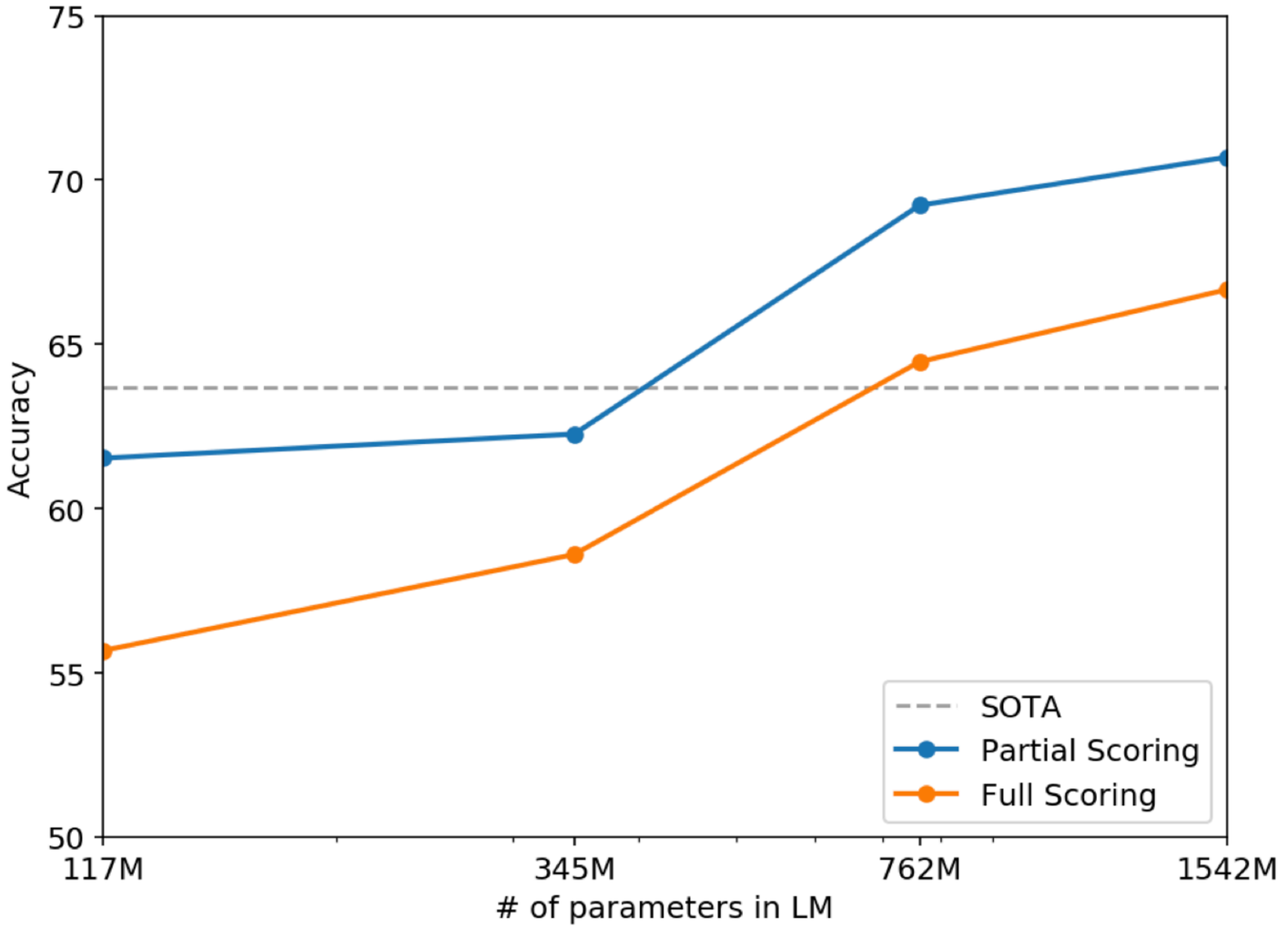


Figure 3. Performance on the Winograd Schema Challenge as a function of model capacity.

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

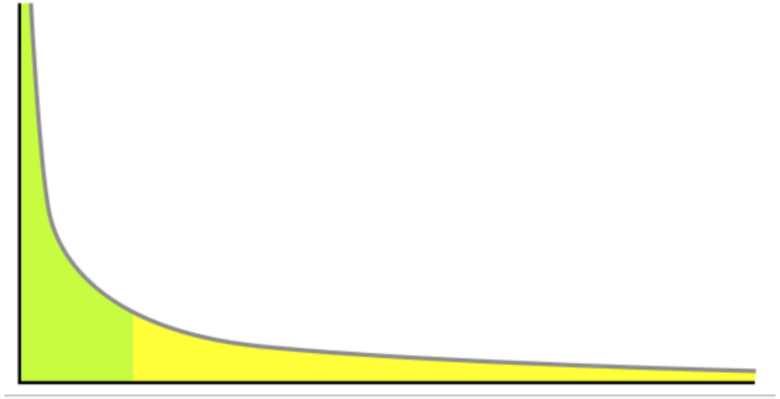
Dario Amodei
OpenAI
damodei@openai.com

https://arxiv.org/abs/2001.08361

Jan 2020

Scaling Laws for LLMs

Power laws



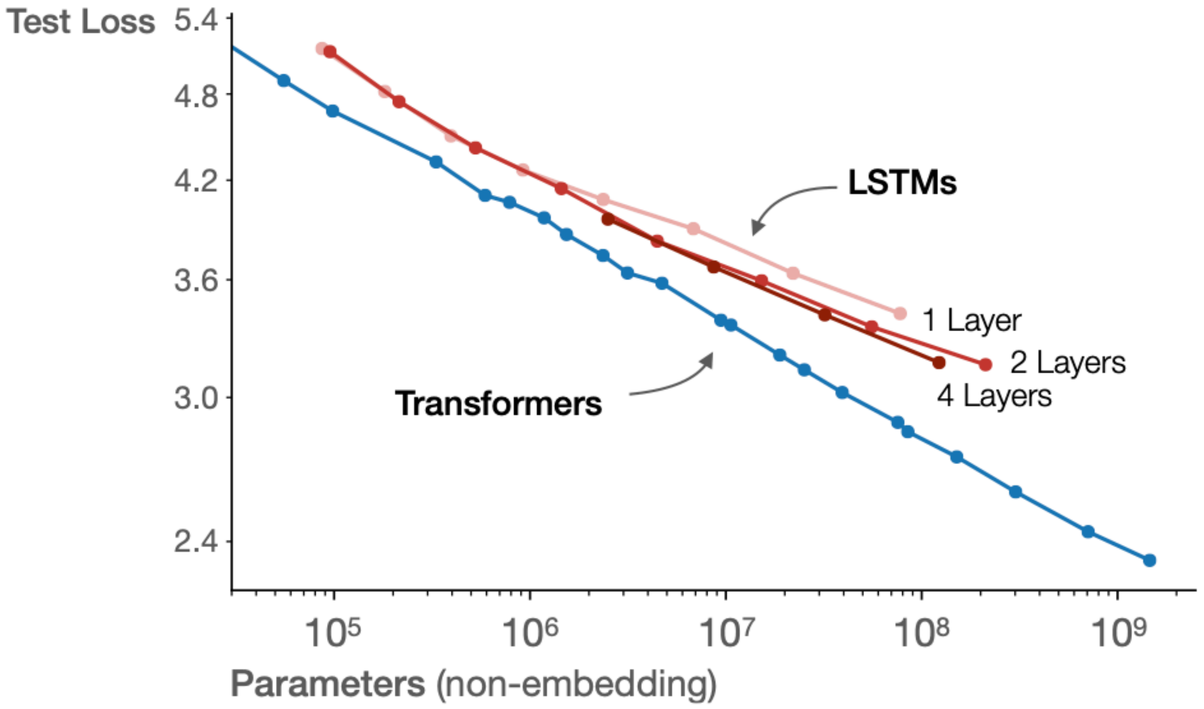
- A power law is a relation between two quantities: $f(x) = (a/x)^k$ e.g. model performance vs. model size.
- Number of model parameters N (excluding subword embeddings)
- Size of dataset D
- Amount of compute (MFLOPs) C
- N, D, C are dominant. Other choices in hyperparameters like width vs. depth are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

https://openai.com/research/ai-and-compute

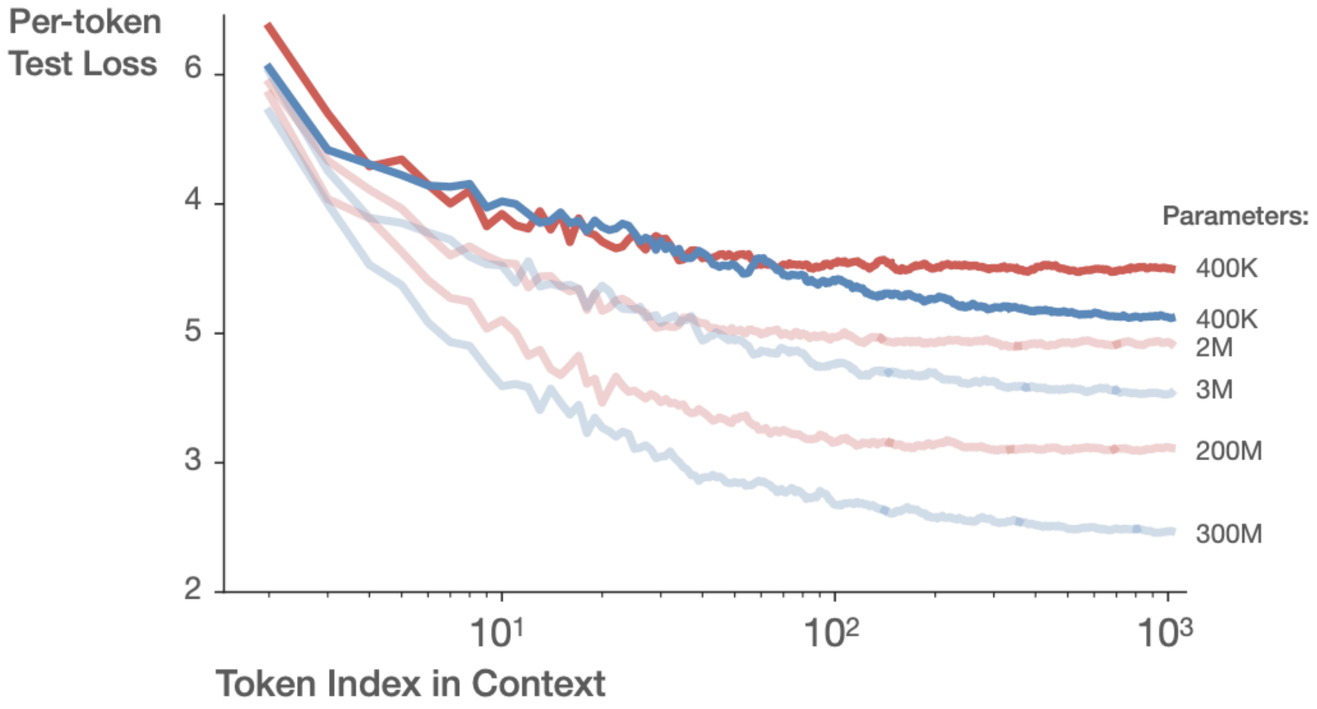
Operation	**Parameters**	**FLOPs per Token**
Embed	$(n_{\text{vocab}} + n_{\text{ctx}})\, d_{\text{model}}$	$4d_{\text{model}}$
Attention: QKV	$n_{\text{layer}} d_{\text{model}} 3d_{\text{attn}}$	$2n_{\text{layer}} d_{\text{model}} 3d_{\text{attn}}$
Attention: Mask	—	$2n_{\text{layer}} n_{\text{ctx}} d_{\text{attn}}$
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	—	$2d_{\text{model}} n_{\text{vocab}}$
Total (Non-Embedding)	$N = 2d_{\text{model}} n_{\text{layer}} (2d_{\text{attn}} + d_{\text{ff}})$	$C_{\text{forward}} = 2N + 2n_{\text{layer}} n_{\text{ctx}} d_{\text{attn}}$

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



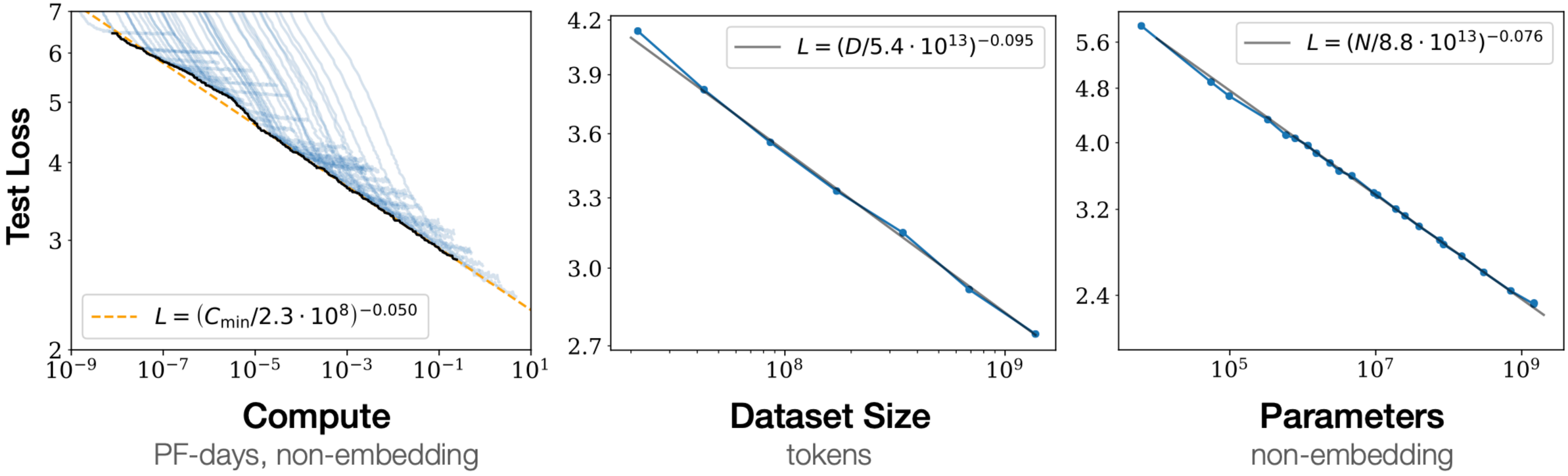


Figure 1 Language modeling performance improves smoothly as we increase the model size, datasetset size, and amount of compute[2] used for training. For optimal performance all three factors must be scaled up in tandem. Empirical performance has a power-law relationship with each individual factor when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

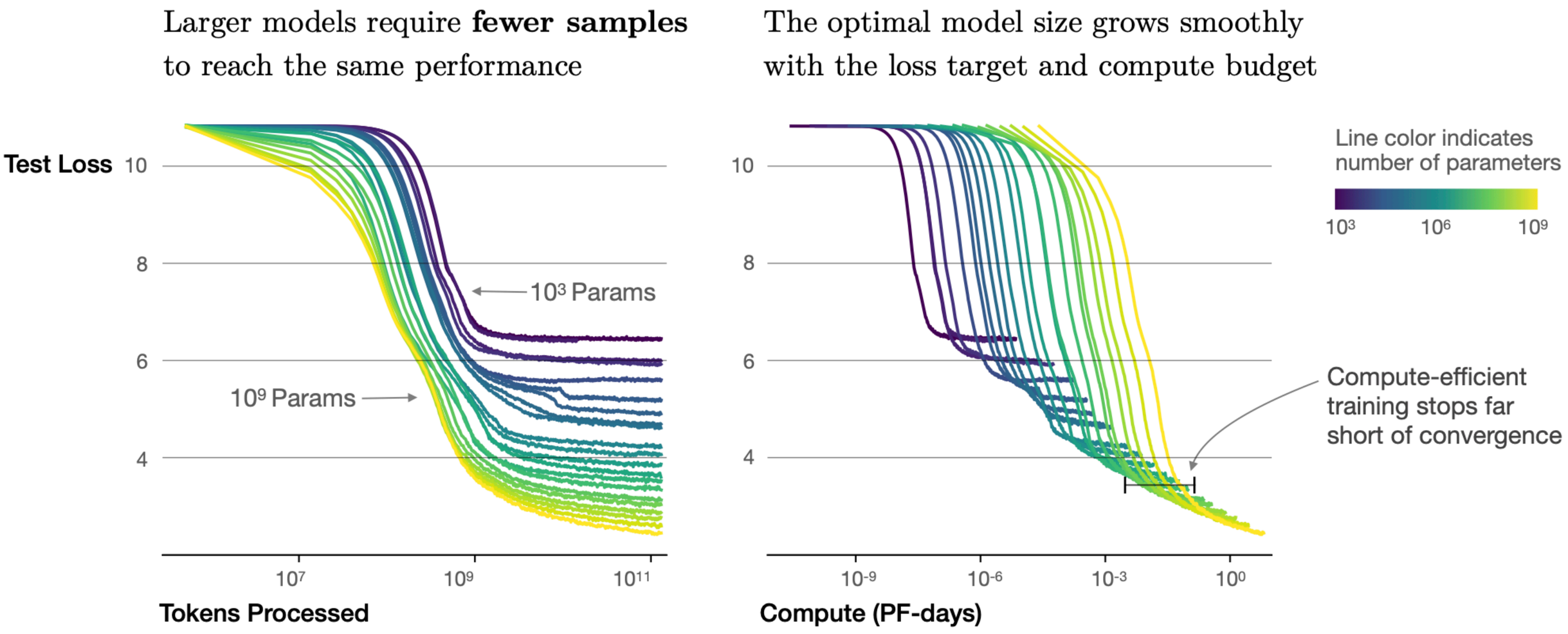


Figure 2 We show a series of language model training runs, with models ranging in size from 10^3 to 10^9 parameters (excluding embeddings).

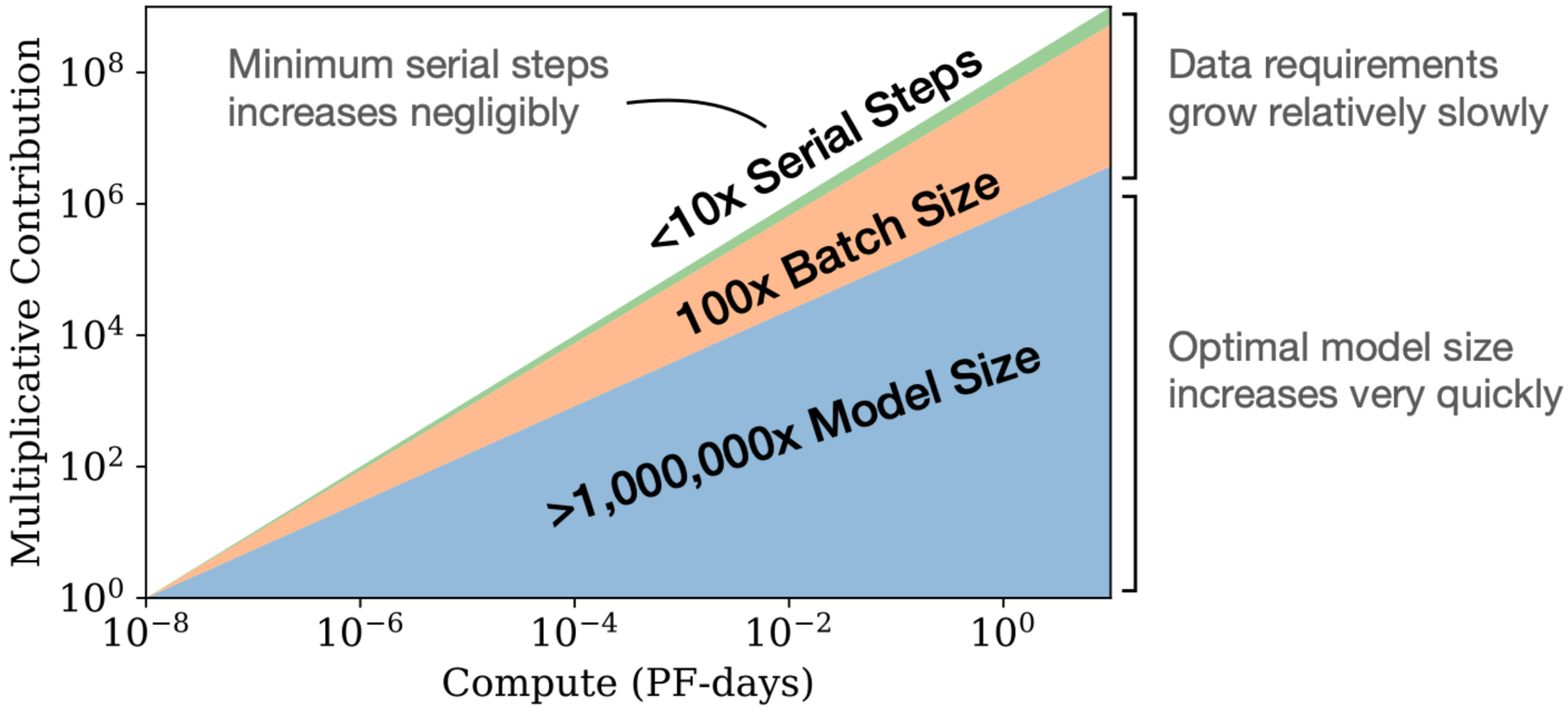
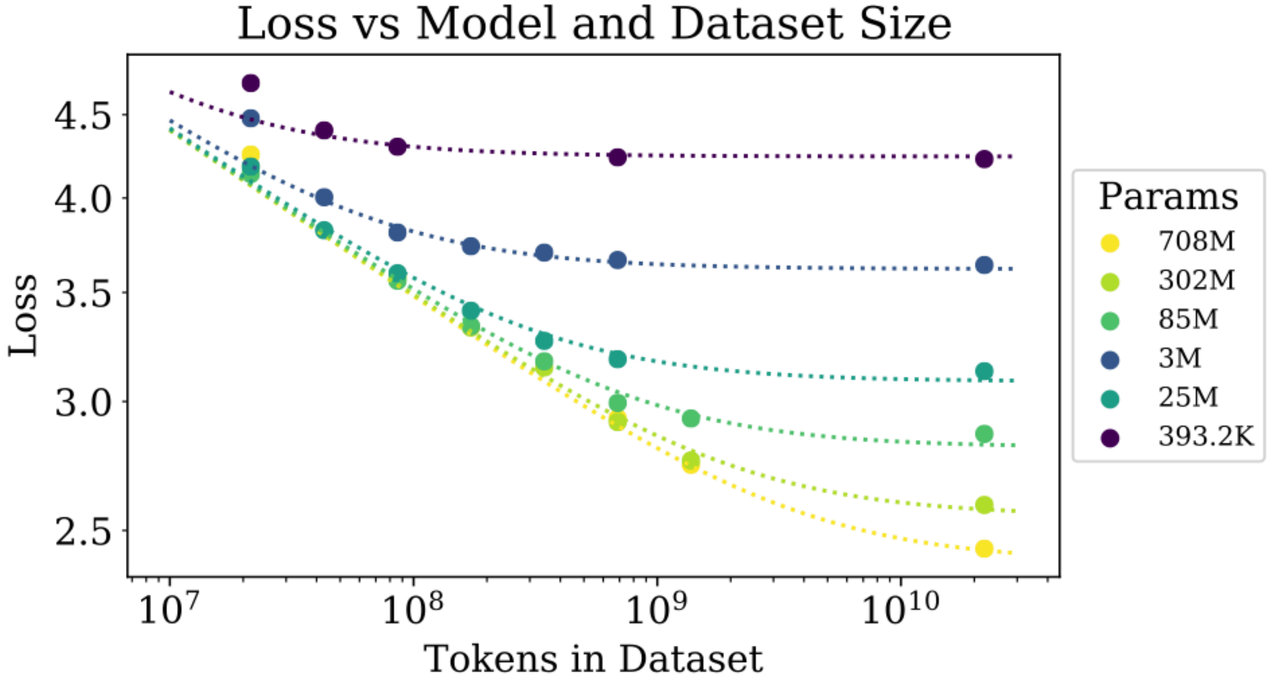
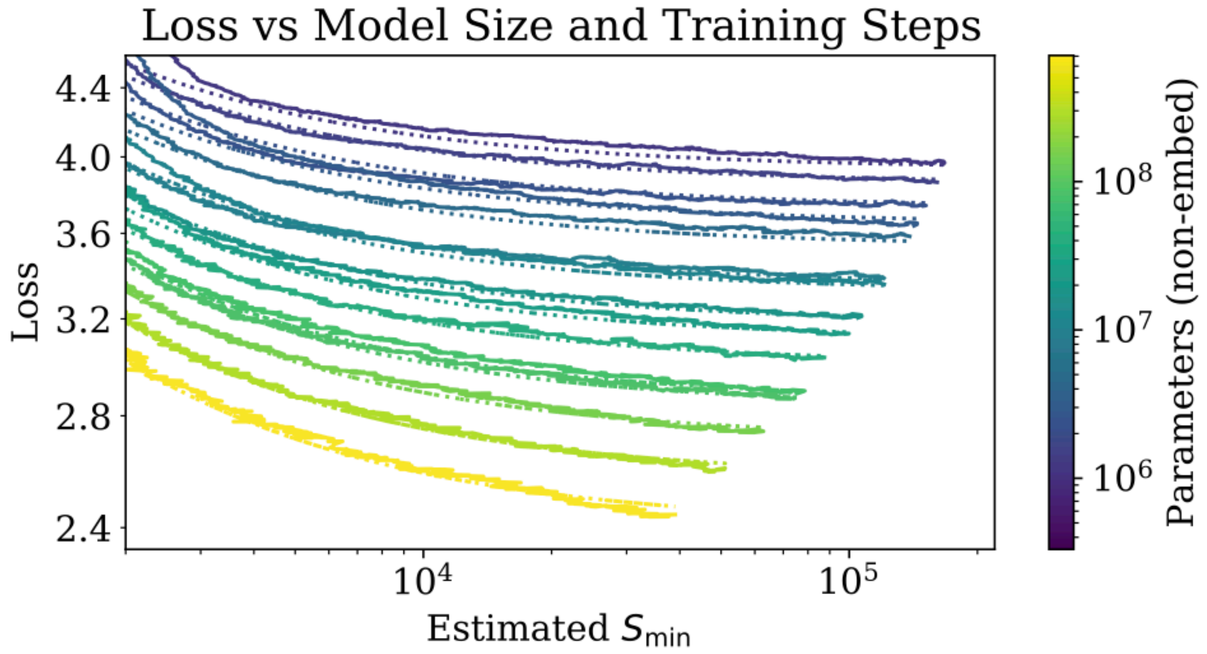


Figure 3 As more compute becomes available, we can choose how much to allocate towards training larger models, using larger batches, and training for more steps. We illustrate this for a billion-fold increase in compute. For optimally compute-efficient training, most of the increase should go towards 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)



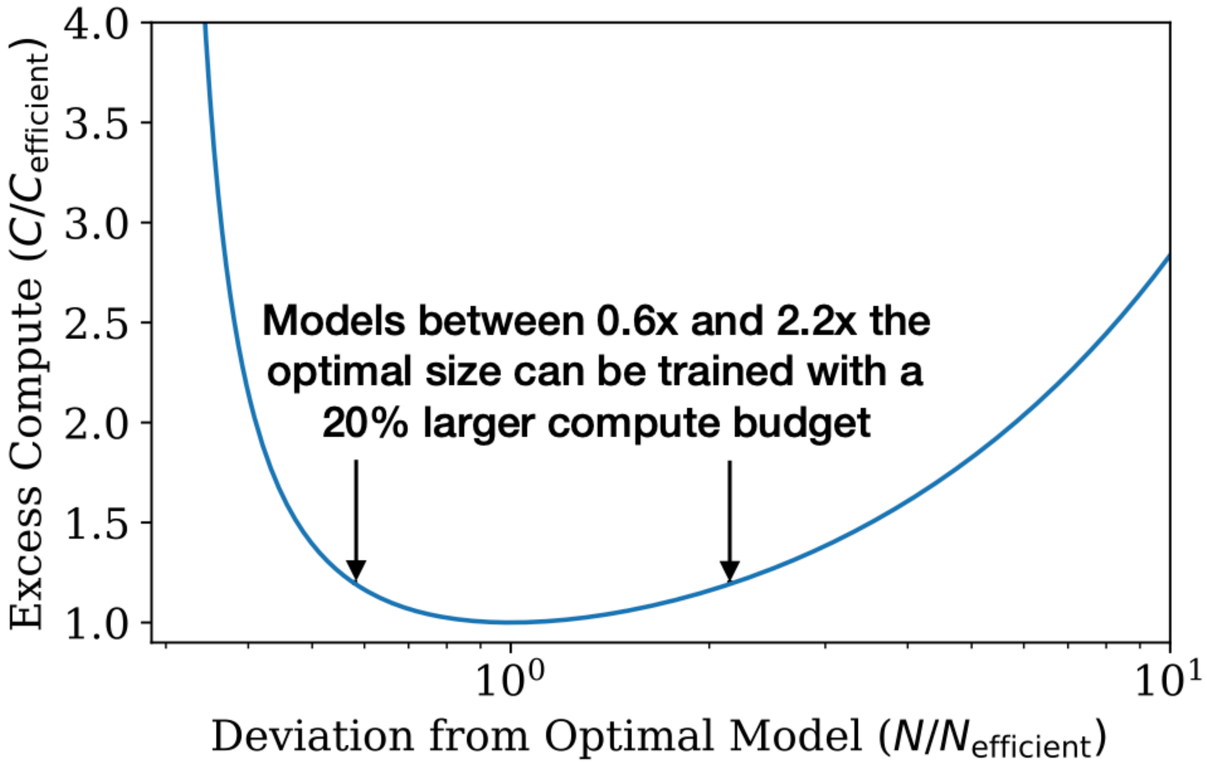


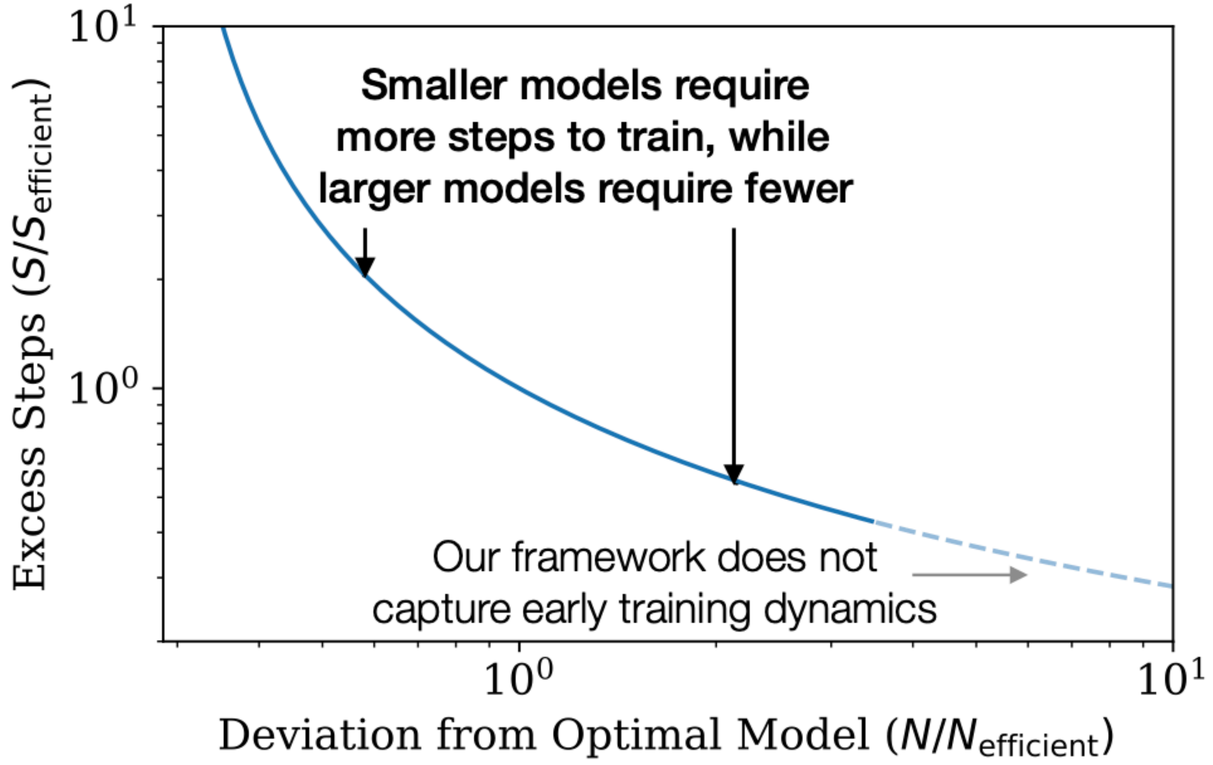
$$L(N, D) = \left[\left(\frac{N_c}{N} \right)^{\frac{\alpha_N}{\alpha_D}} + \frac{D_c}{D} \right]^{\alpha_D}$$

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

S = parameter update steps

Optimal Allocation of Compute Budget





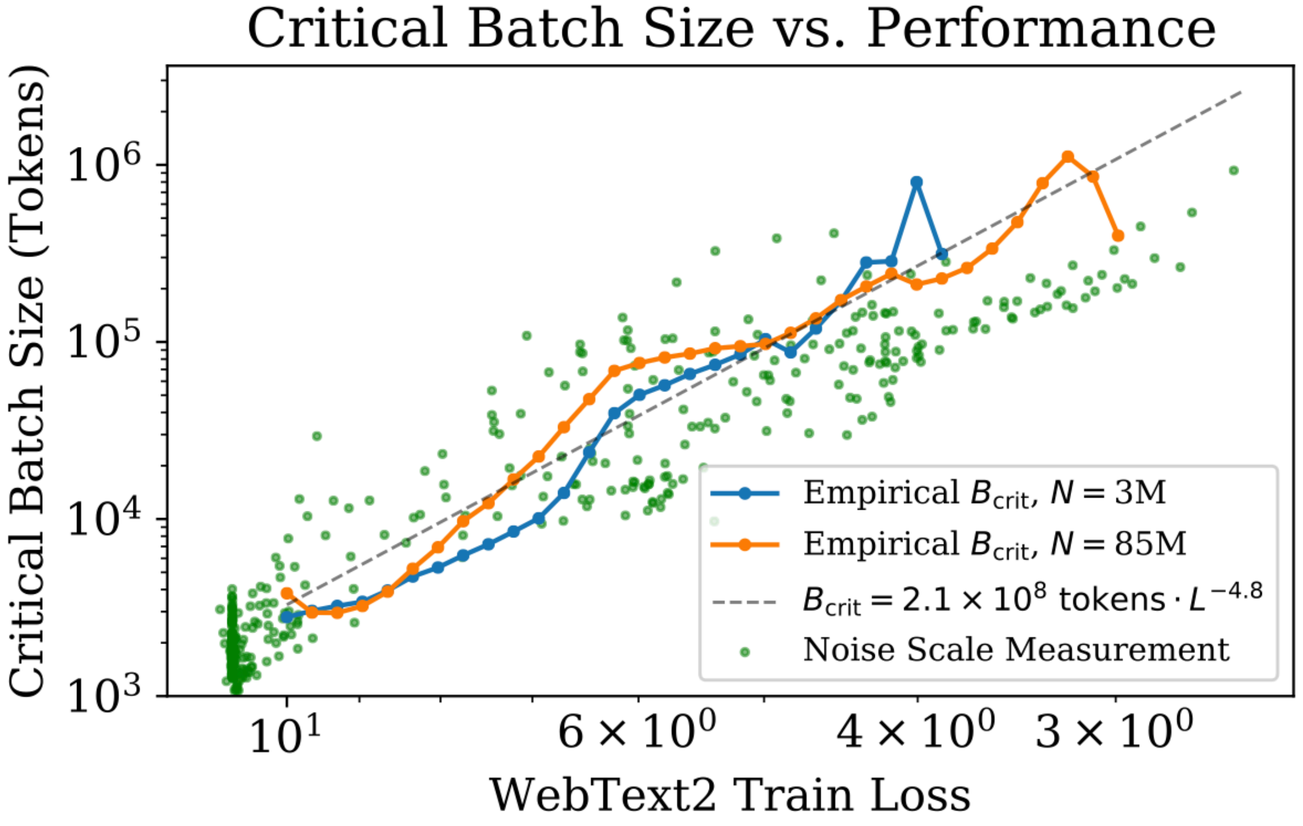


Figure 10 The critical batch size B_{crit} follows a power law in the loss as performance increase, and does not depend directly on the model size. We find that the critical batch size approximately doubles for every 13% decrease in loss. B_{crit} is measured empirically from the data shown in Figure 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

DeepMind

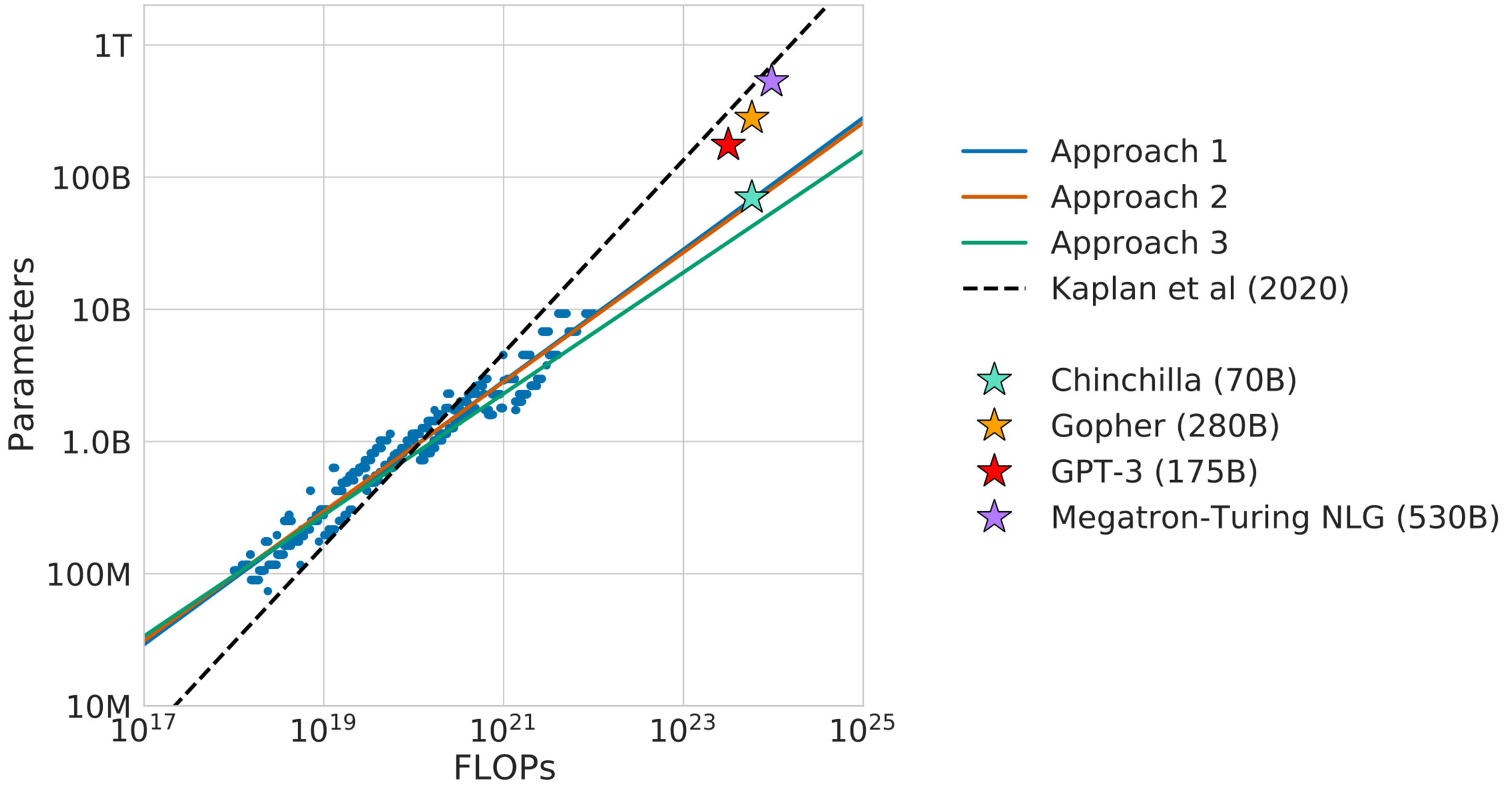
Training Compute-Optimal Large Language Models

Jordan Hoffmann*, Sebastian Borgeaud*, Arthur Mensch*, 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*

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**.


1T
100B
10B
1.0B
100M
10M
Parameters
10^{17}
10^{19}
10^{21}
10^{23}
10^{25}
FLOPs
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 (Thoppilan et al., 2022)	137 Billion	168 Billion
GPT-3 (Brown et al., 2020)	175 Billion	300 Billion
Jurassic (Lieber et al., 2021)	178 Billion	300 Billion
Gopher (Rae et al., 2021)	280 Billion	300 Billion
MT-NLG 530B (Smith et al., 2022)	530 Billion	270 Billion
Chinchilla	70 Billion	1.4 Trillion

The GPT3 paper

Language Models are Few-Shot Learners

Tom B. Brown* **Benjamin Mann*** **Nick Ryder*** **Melanie Subbiah***

Jared Kaplan† **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**

https://arxiv.org/abs/2005.14165

NeurIPS 2020, Vancouver, BC

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.

```
1  Translate English to French:        ← task description
2  cheese =>                           ← prompt
```

One-shot

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

```
1  Translate English to French:        ← task description
2  sea otter => loutre de mer          ← example
3  cheese =>                           ← prompt
```

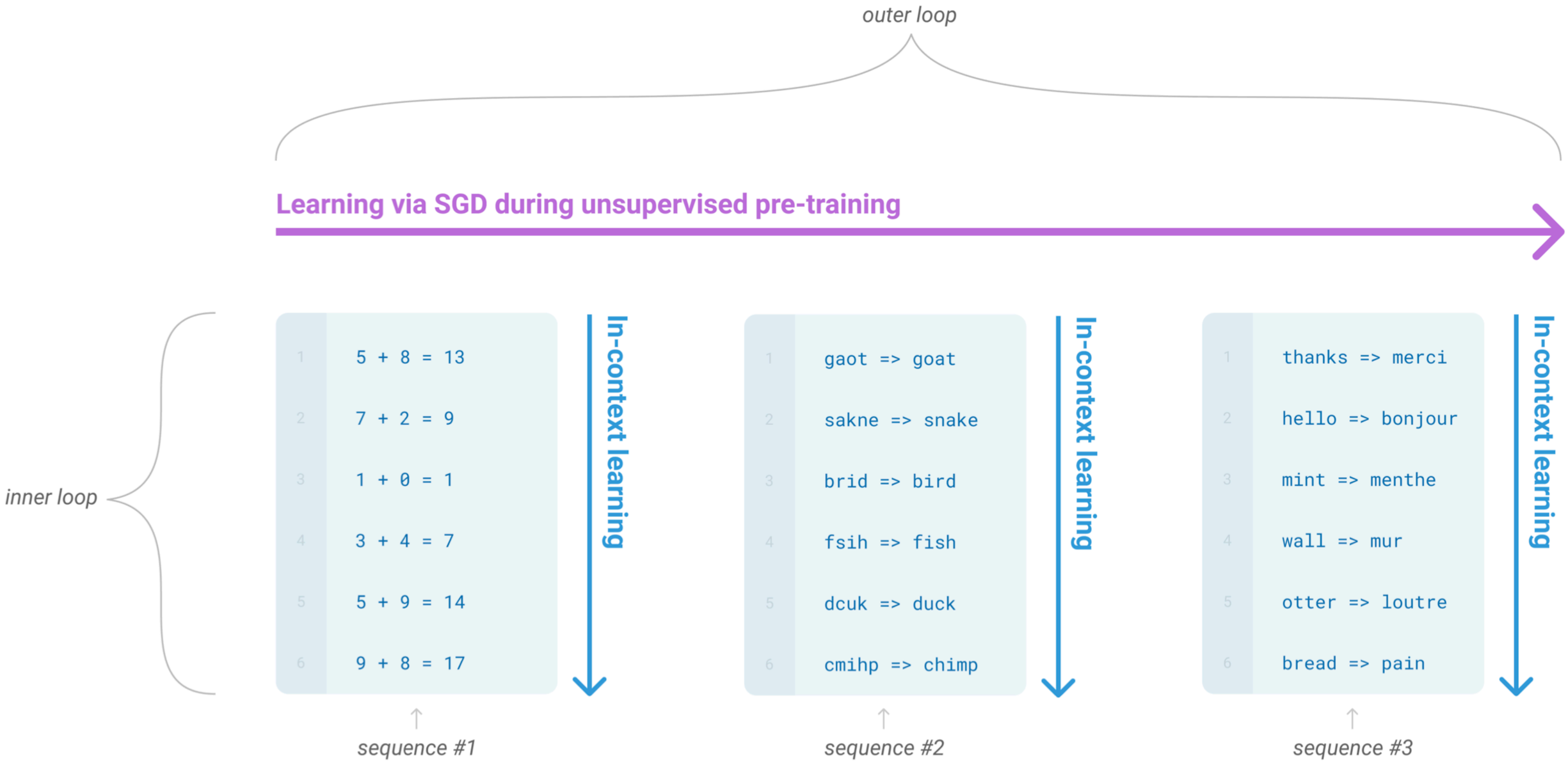
Few-shot

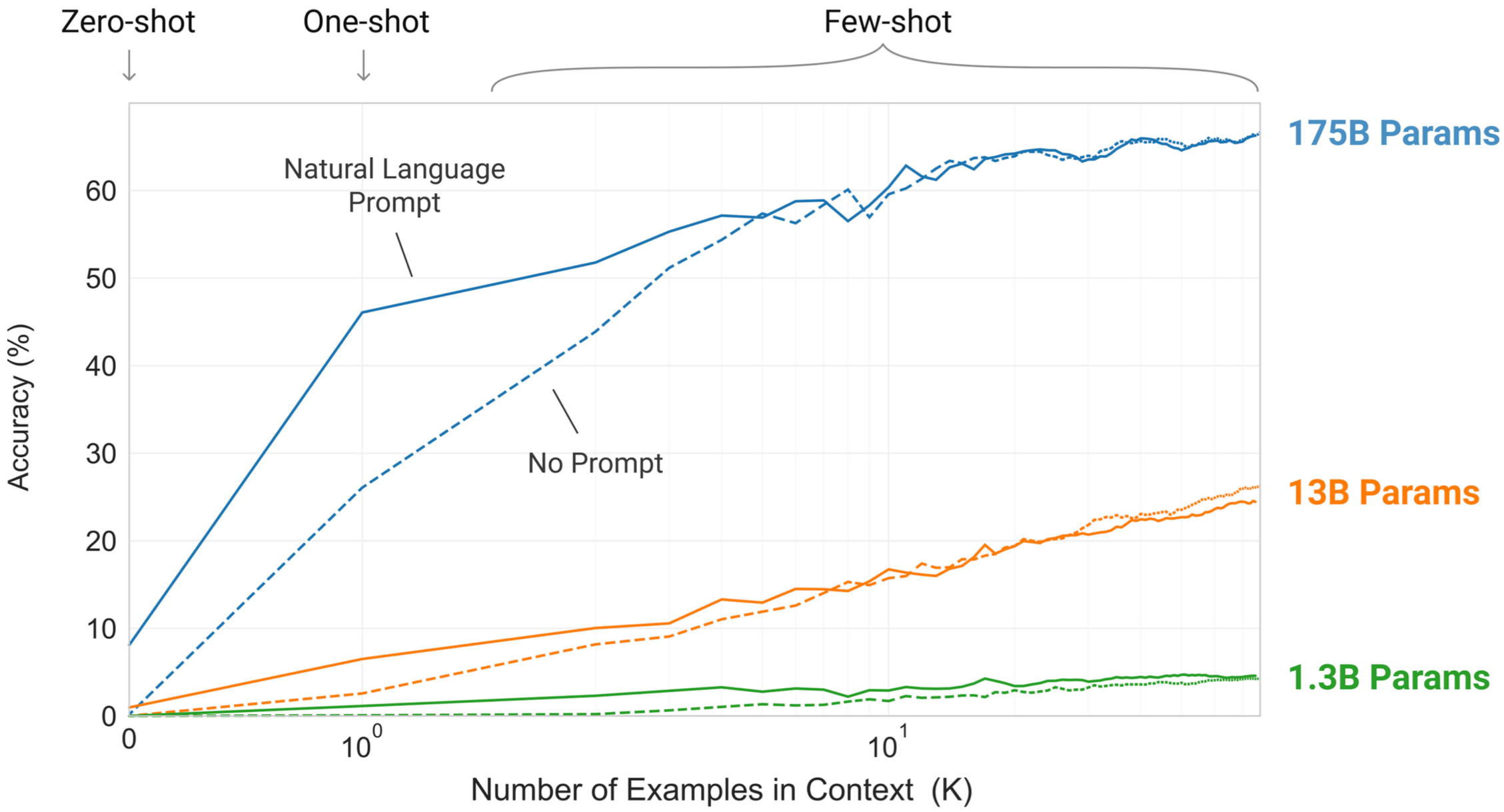
In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

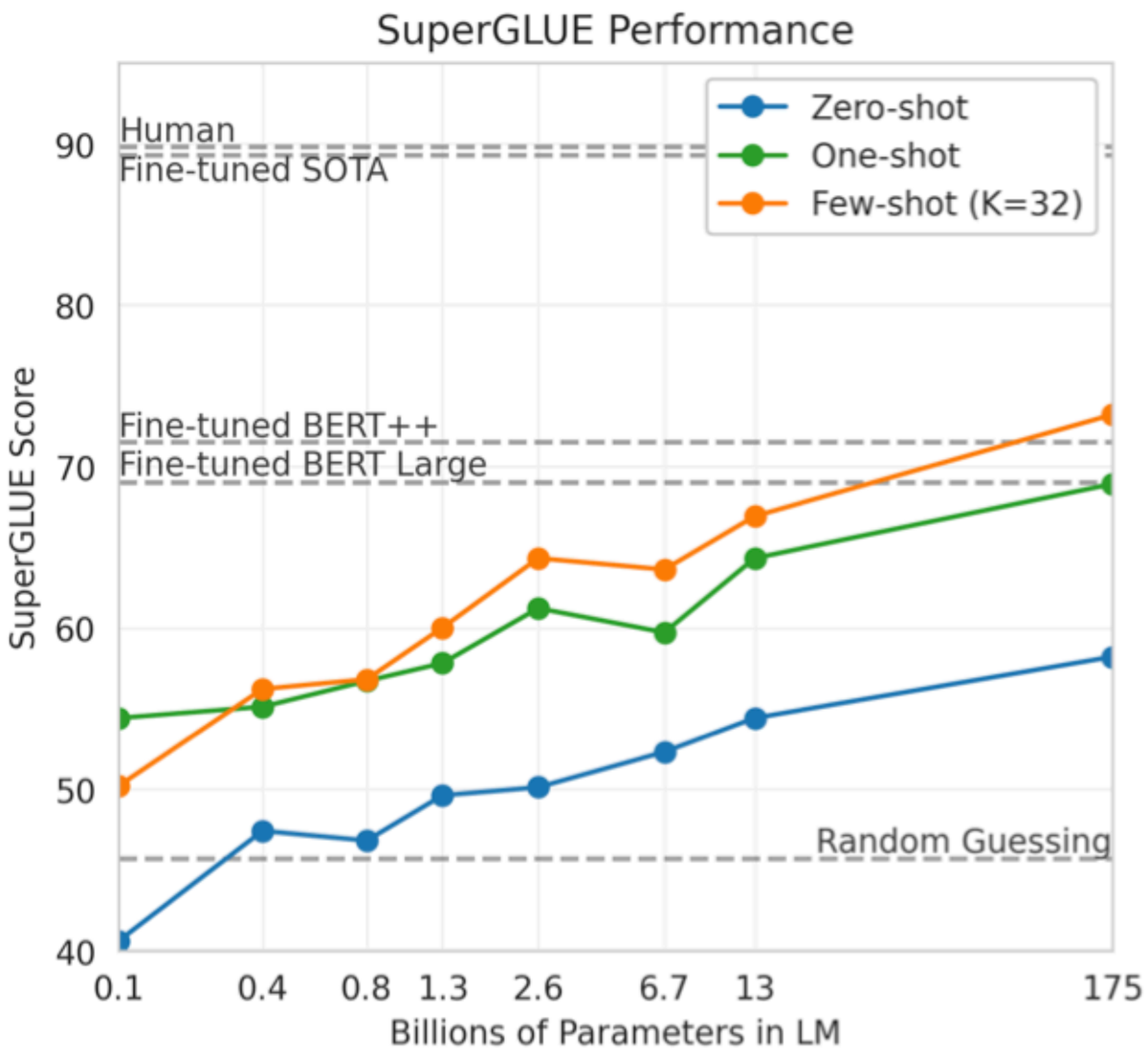
```
1  Translate English to French:        ← task description
2  sea otter => loutre de mer          ← examples
3  peppermint => menthe poivrée        ←
4  plush girafe => girafe peluche      ←
5  cheese =>                           ← prompt
```

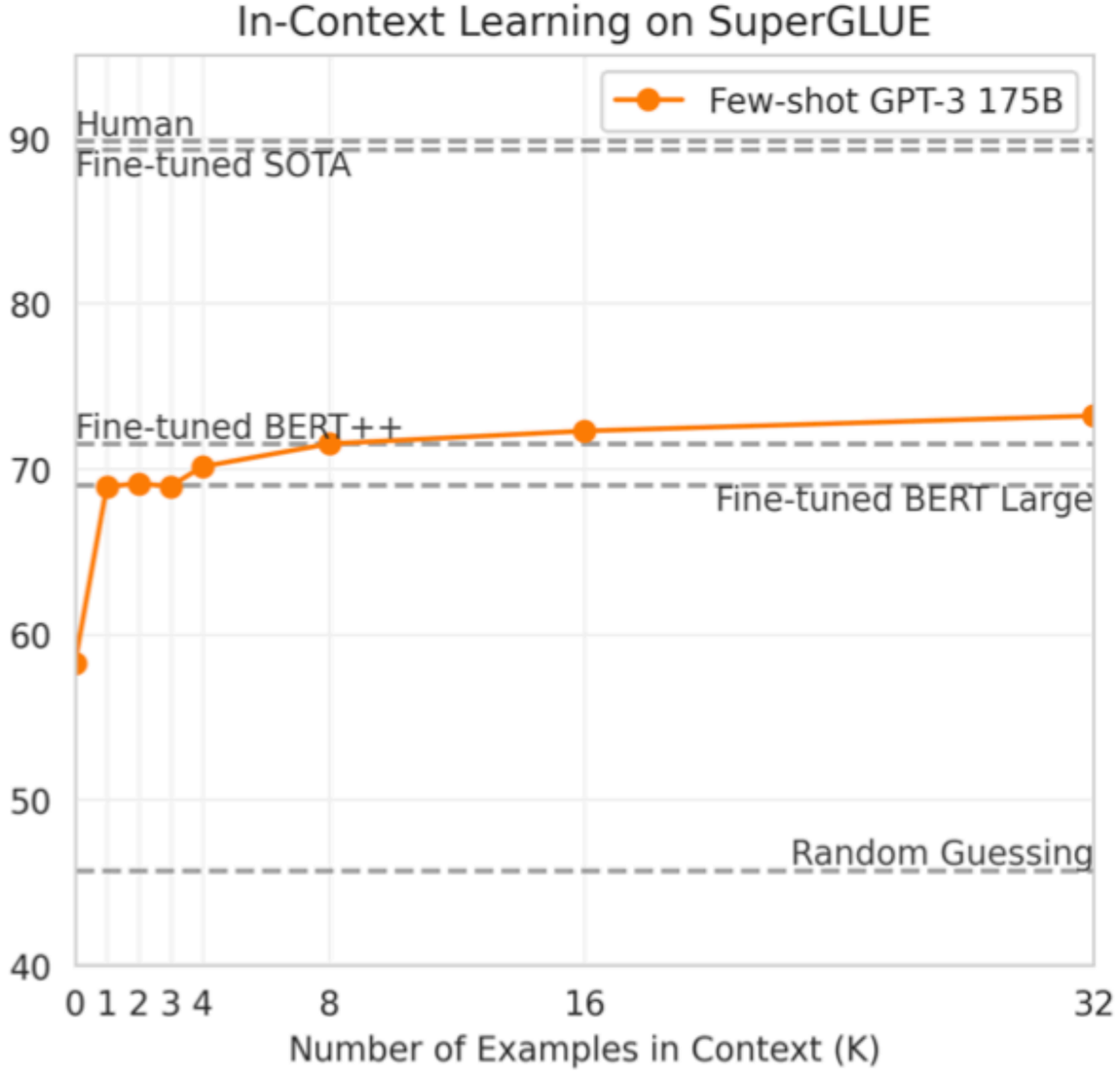
Fine-tuning fails at scale

- 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
In-context learning
sequence #1
1 gaot => goat
2 sakne => snake
3 brid => bird
4 fsih => fish
5 dcuk => duck
6 cmihp => chimp
In-context learning
sequence #2
1 thanks => merci
2 hello => bonjour
3 mint => menthe
4 wall => mur
5 otter => loutre
6 bread => pain
In-context learning
sequence #3


Zero-shot
One-shot
Few-shot
Natural Language Prompt
No Prompt
175B Params
13B Params
1.3B Params
Accuracy (%)
0
10
20
30
40
50
60
0
10^0
10^1
Number of Examples in Context (K)





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	**89.0**	**91.0**	**96.9**	**93.9**	**94.8**	**92.5**
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	**76.1**	**93.8**	**62.3**	**88.2**	**92.5**	**93.3**
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.

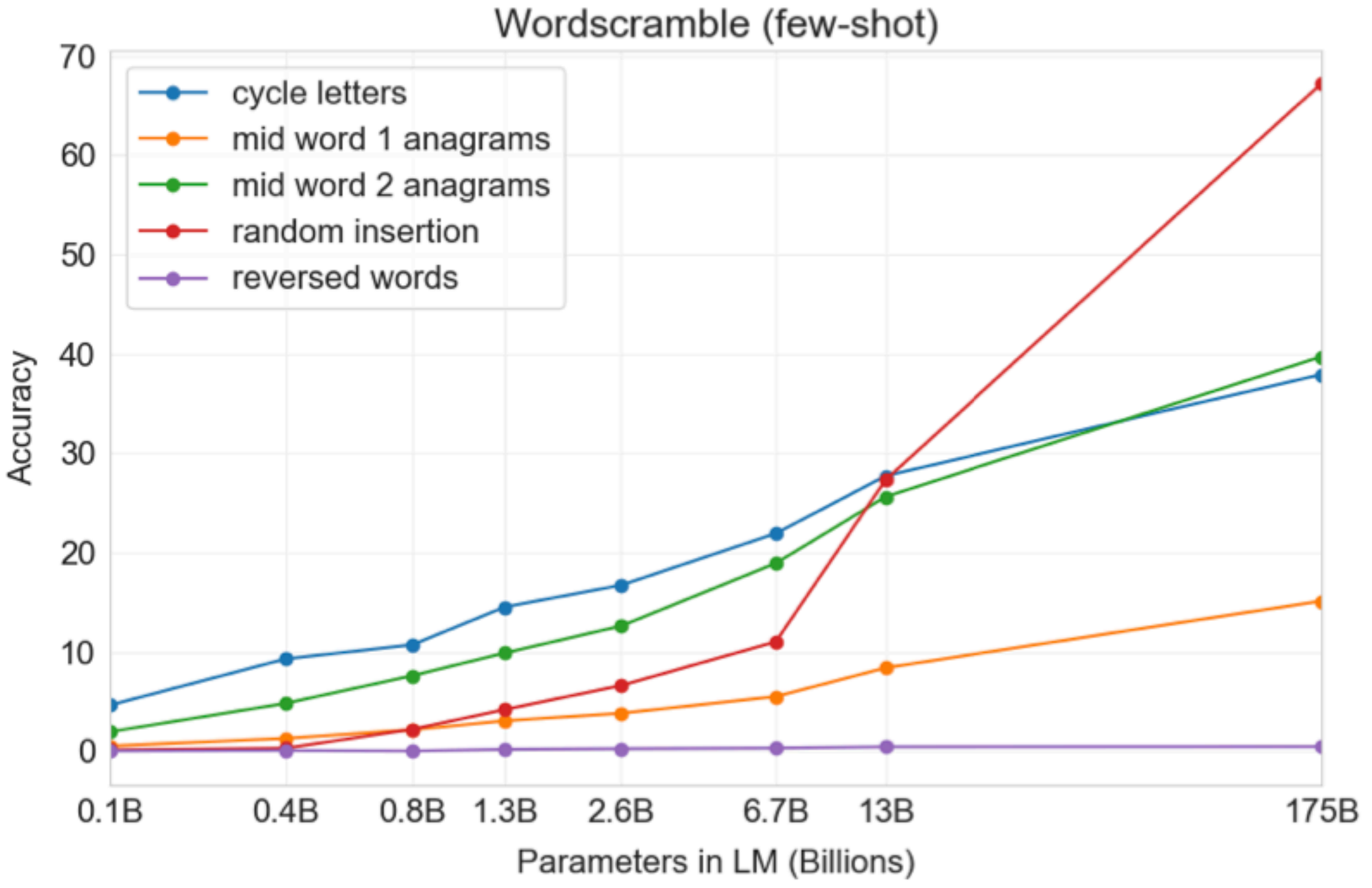
Setting	LAMBADA (acc)	LAMBADA (ppl)	StoryCloze (acc)	HellaSwag (acc)
SOTA	68.0[a]	8.63[b]	**91.8**[c]	**85.6**[d]
GPT-3 Zero-Shot	**76.2**	**3.00**	83.2	78.9
GPT-3 One-Shot	**72.5**	**3.35**	84.7	78.1
GPT-3 Few-Shot	**86.4**	**1.92**	87.7	79.3

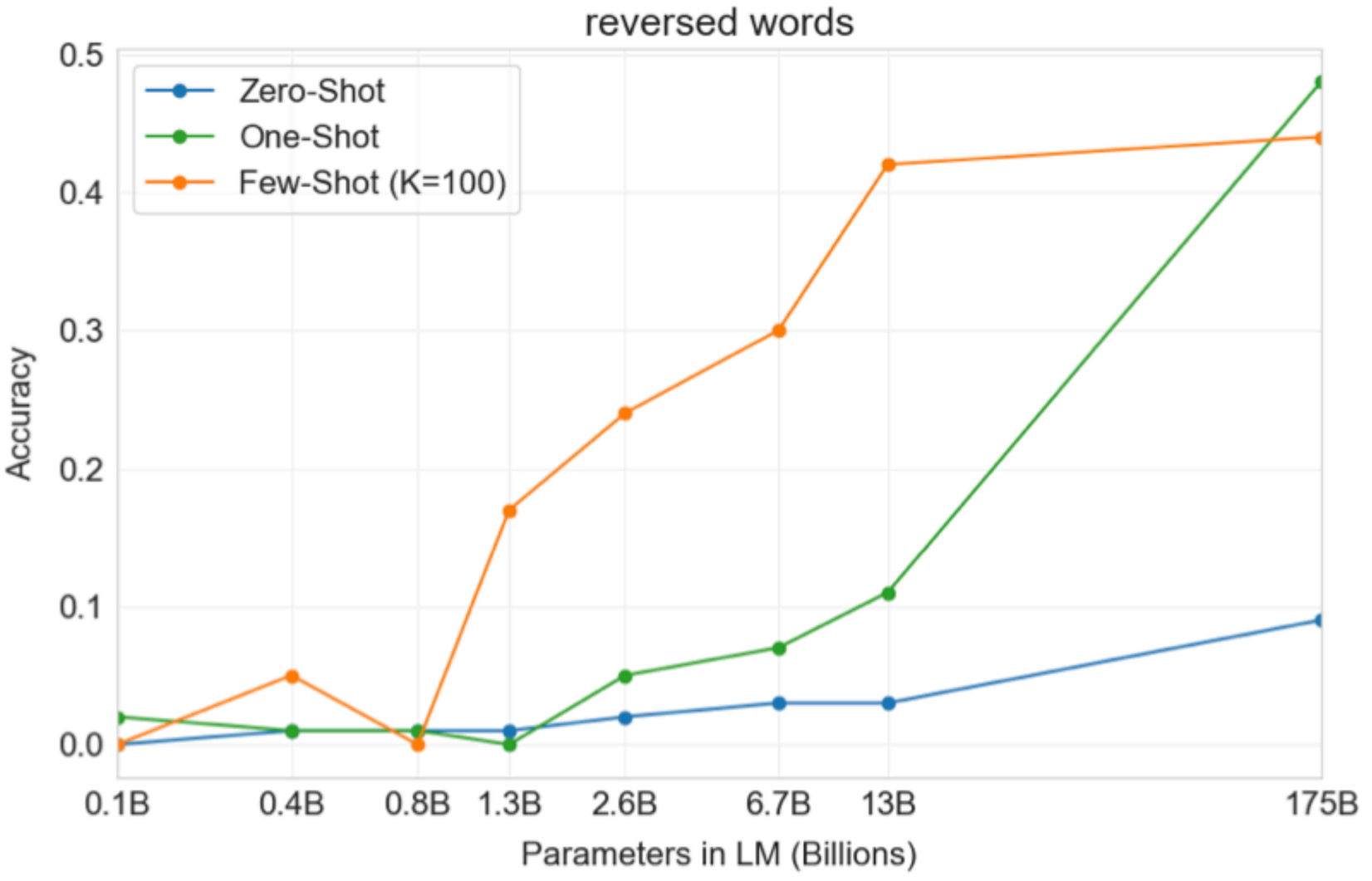
Setting	NaturalQS	WebQS	TriviaQA
RAG (Fine-tuned, Open-Domain) [LPP+20]	**44.5**	**45.5**	**68.0**
T5-11B+SSM (Fine-tuned, Closed-Book) [RRS20]	36.6	44.7	60.5
T5-11B (Fine-tuned, Closed-Book)	34.5	37.4	50.1
GPT-3 Zero-Shot	14.6	14.4	64.3
GPT-3 One-Shot	23.0	25.3	**68.0**
GPT-3 Few-Shot	29.9	41.5	**71.2**

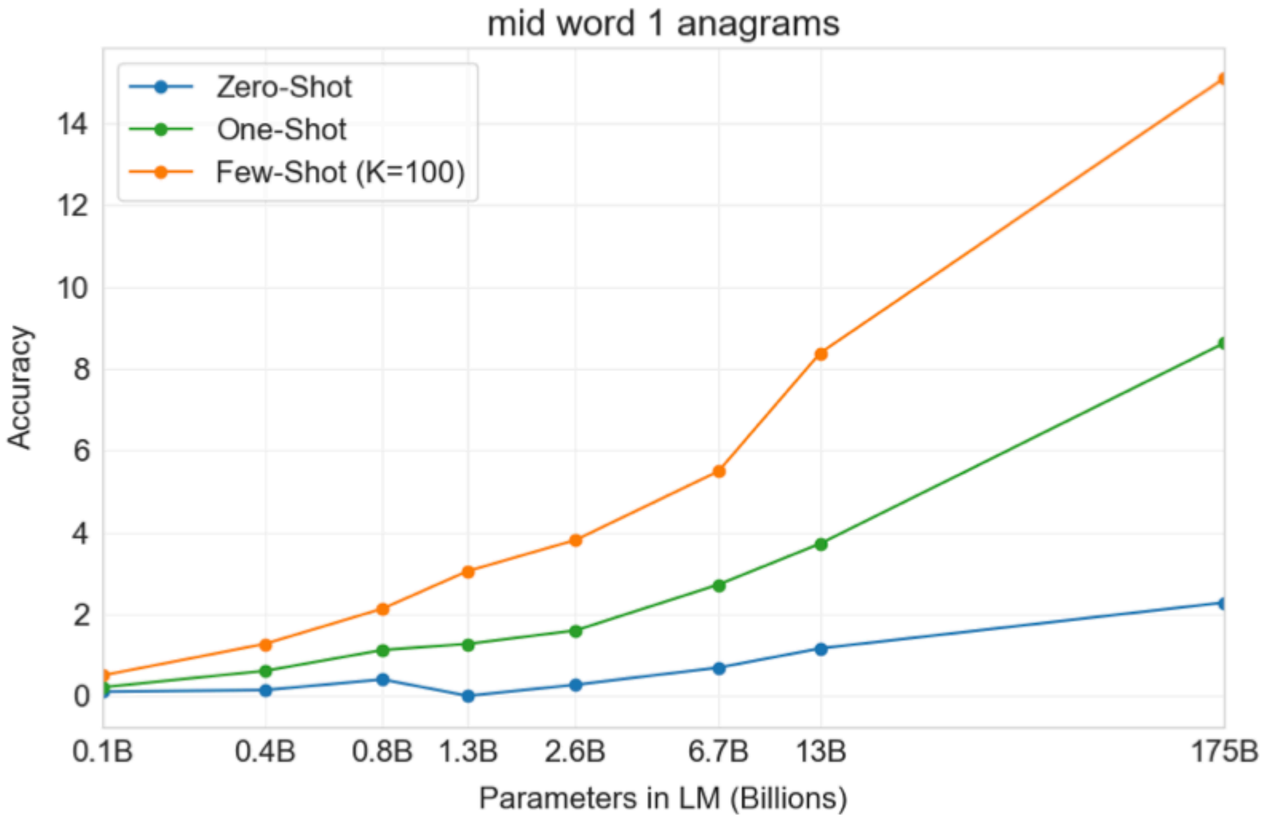
Setting	ARC (Easy)	ARC (Challenge)	CoQA	DROP
Fine-tuned SOTA	**92.0**[a]	**78.5**[b]	**90.7**[c]	**89.1**[d]
GPT-3 Zero-Shot	68.8	51.4	81.5	23.6
GPT-3 One-Shot	71.2	53.2	84.0	34.3
GPT-3 Few-Shot	70.1	51.5	85.0	36.5

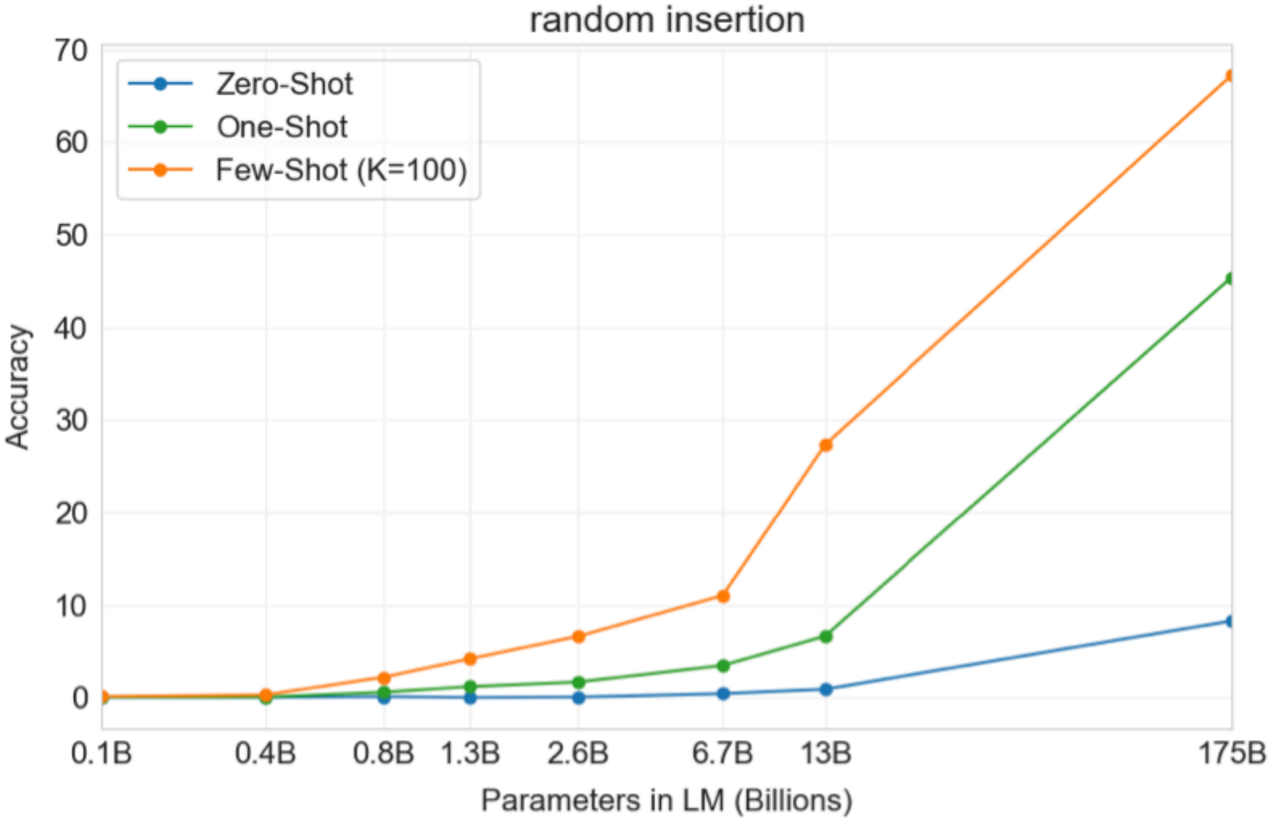
WMT 2014

Setting	En→Fr	Fr→En	En→De	De→En	En→Ro	Ro→En
SOTA (Supervised)	**45.6**[a]	35.0 [b]	**41.2**[c]	40.2[d]	**38.5**[e]	**39.9**[e]
XLM [LC19]	33.4	33.3	26.4	34.3	33.3	31.8
MASS [STQ+19]	37.5	34.9	28.3	35.2	35.2	33.1
mBART [LGG+20]	-	-	29.8	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	39.2	29.7	40.6	21.0	39.5


Wordscramble (few-shot)
cycle letters
mid word 1 anagrams
mid word 2 anagrams
random insertion
reversed words
Accuracy
70
60
50
40
30
20
10
0
0.1B
0.4B
0.8B
1.3B
2.6B
6.7B
13B
175B
Parameters in LM (Billions)


reversed words
Zero-Shot
One-Shot
Few-Shot (K=100)
Accuracy
0.5
0.4
0.3
0.2
0.1
0.0
0.1B
0.4B
0.8B
1.3B
2.6B
6.7B
13B
175B
Parameters in LM (Billions)


mid word 1 anagrams
Zero-Shot
One-Shot
Few-Shot (K=100)
Accuracy
0
2
4
6
8
10
12
14
0.1B
0.4B
0.8B
1.3B
2.6B
6.7B
13B
175B
Parameters in LM (Billions)


random insertion
Zero-Shot
One-Shot
Few-Shot (K=100)
Accuracy
0
10
20
30
40
50
60
70
0.1B
0.4B
0.8B
1.3B
2.6B
6.7B
13B
175B
Parameters in LM (Billions)

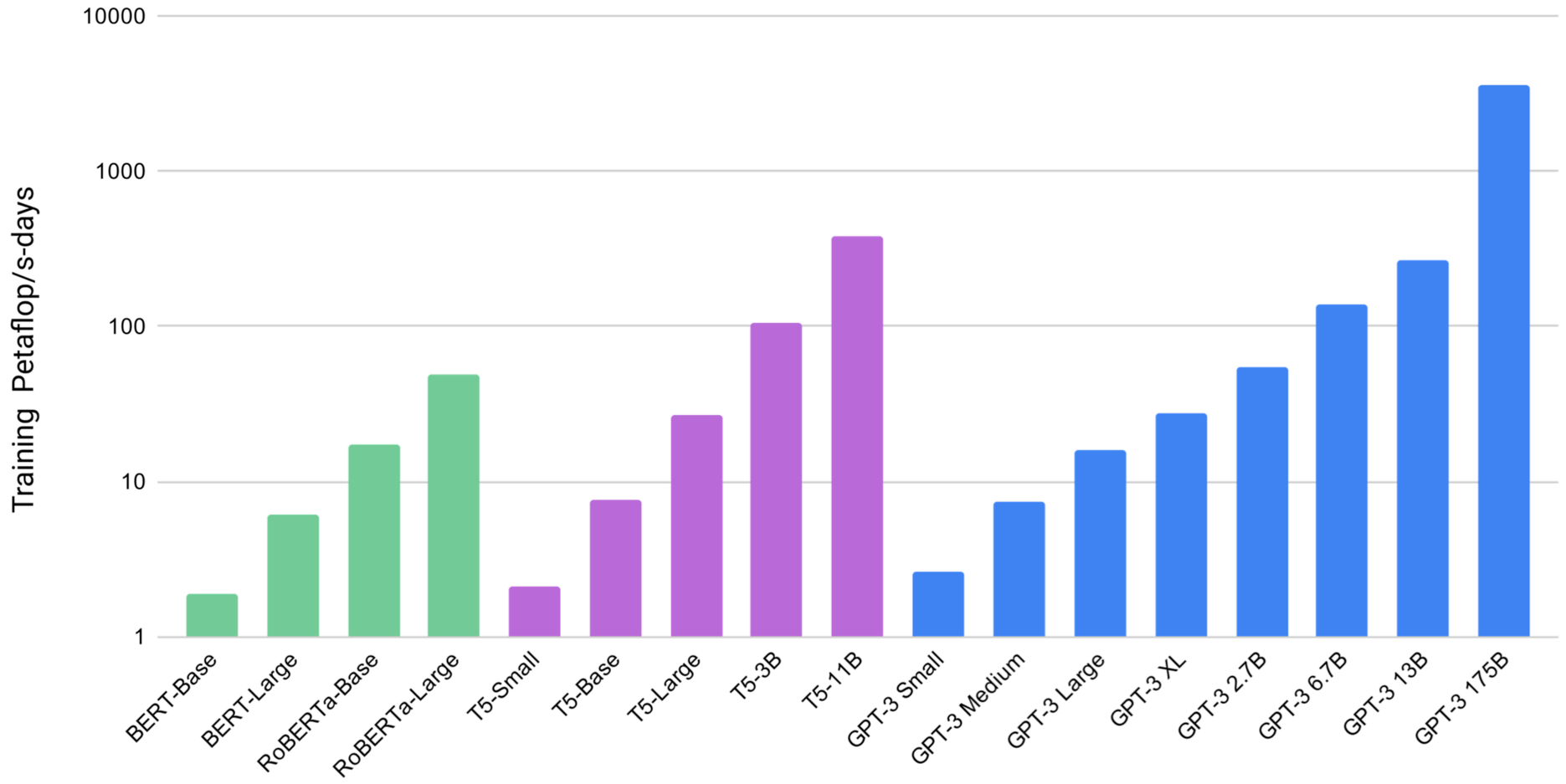
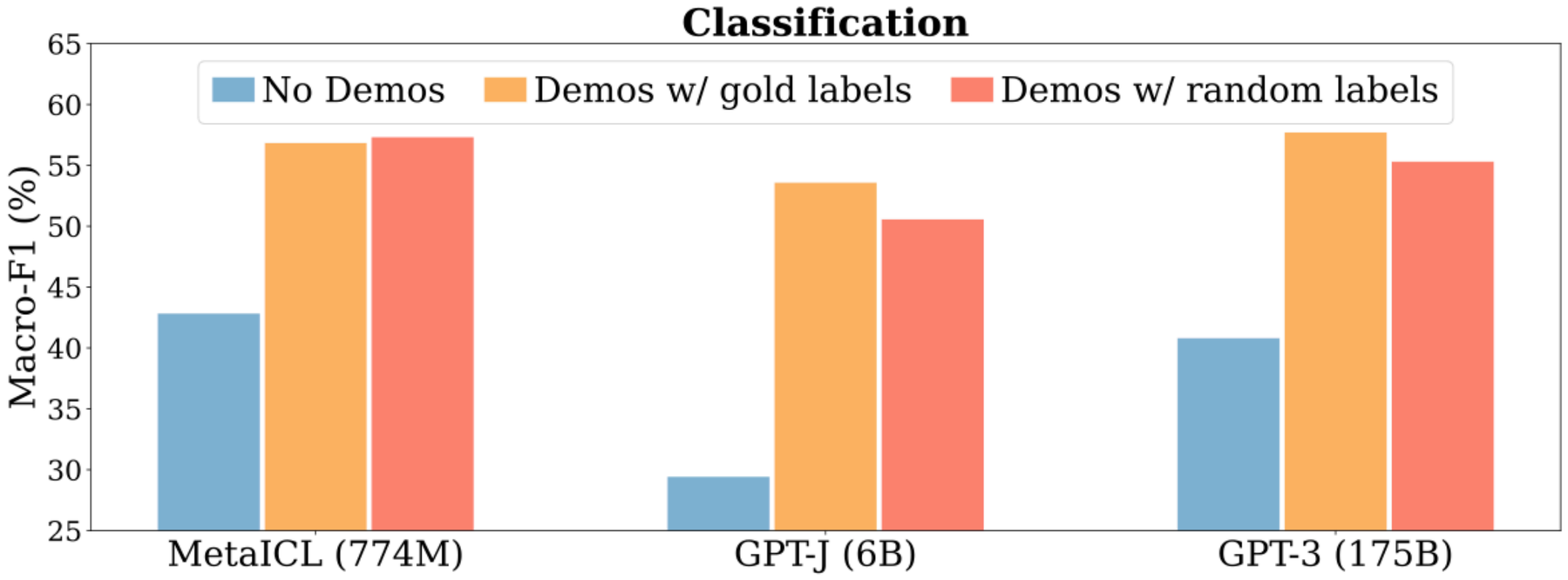
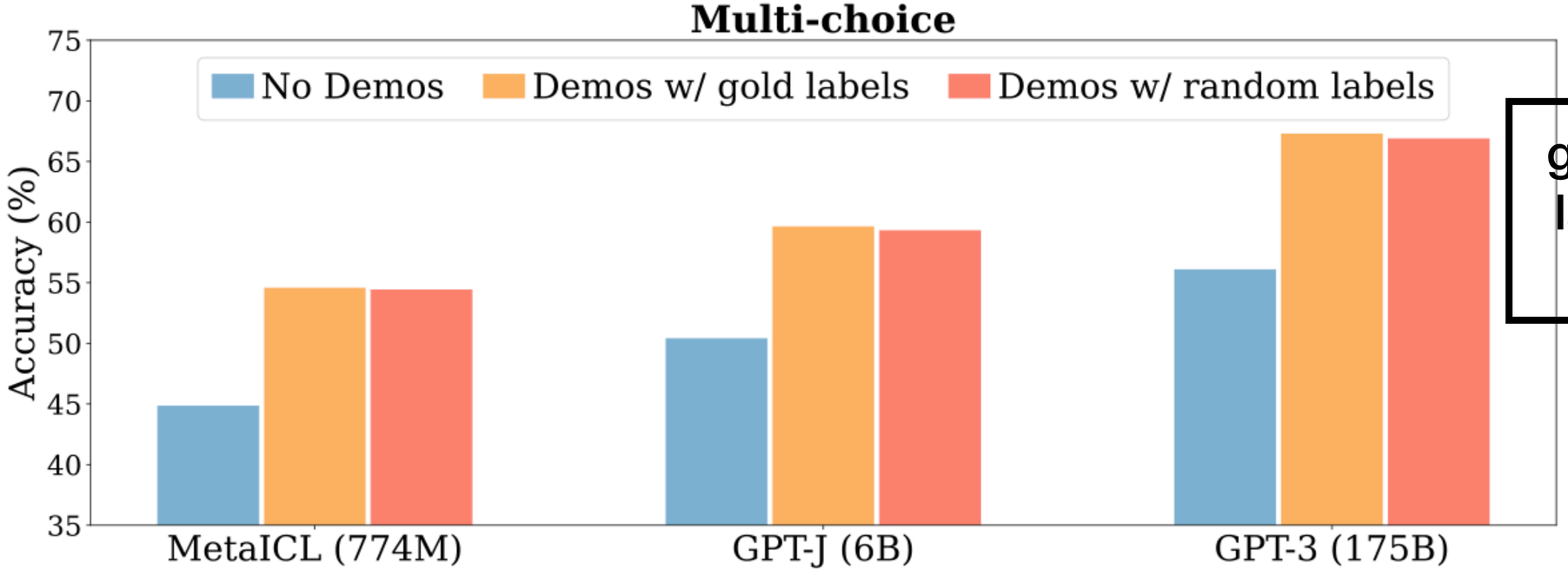


Figure 7.2: Total compute used during training. Based on the analysis in Scaling Laws For Neural Language Models [KMH+20] we train much larger models on many fewer tokens than is typical. As a consequence, although GPT-3 3B is almost 10x larger than RoBERTa-Large (355M params), both models took roughly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

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

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





ground truth labels don't matter!

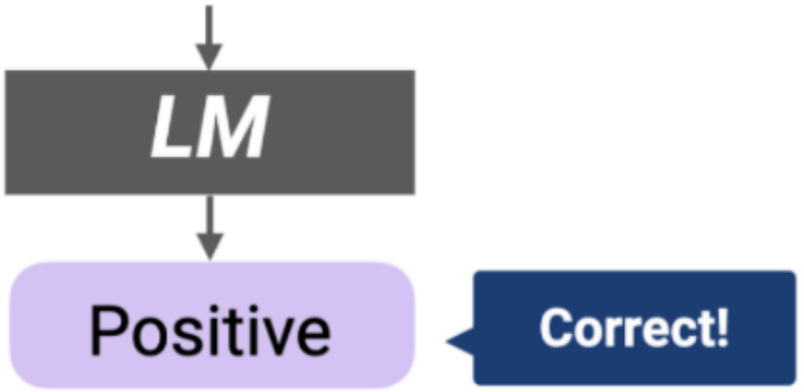
ground truth labels

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 _____



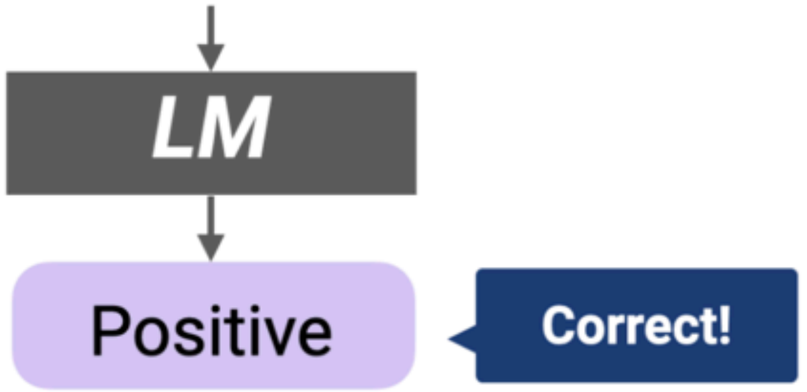
replace true labels with random labels

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?

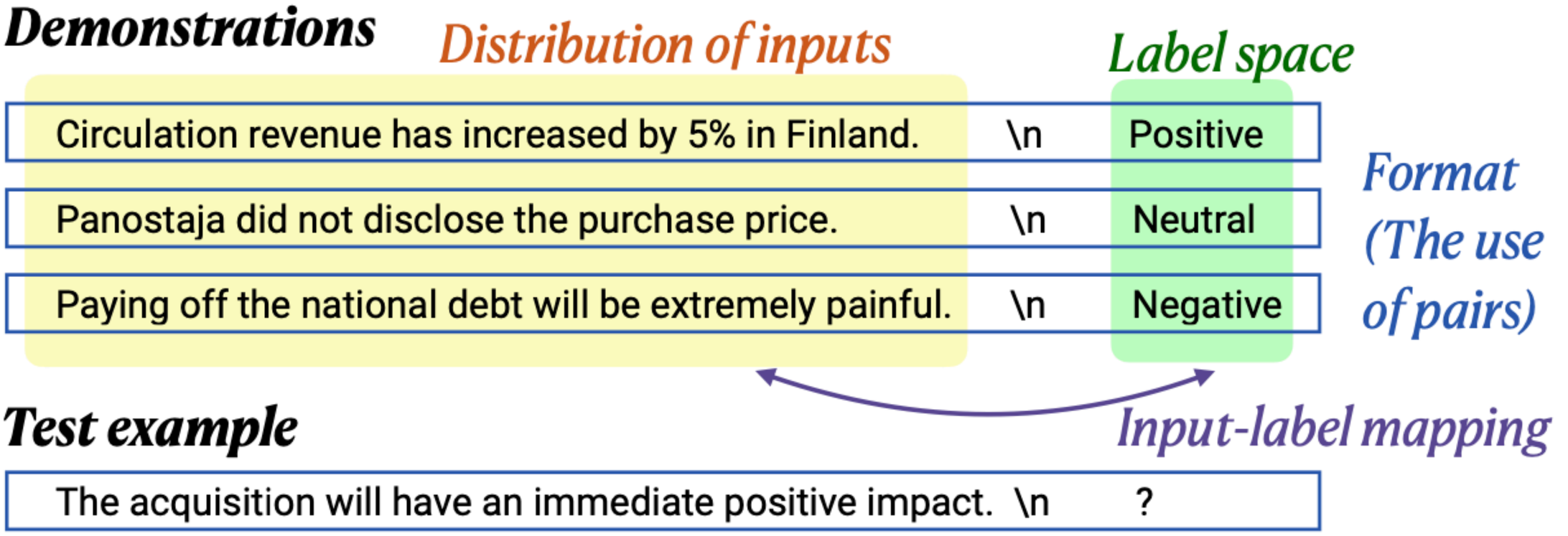
Four hypotheses

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 $x_1, \ldots, x_k$ are from (is it from a sports article, or business news?)

3. The output label space $y_1, \ldots, y_k$

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)
Input-label mapping
Test example
The acquisition will have an immediate positive impact. \n ?

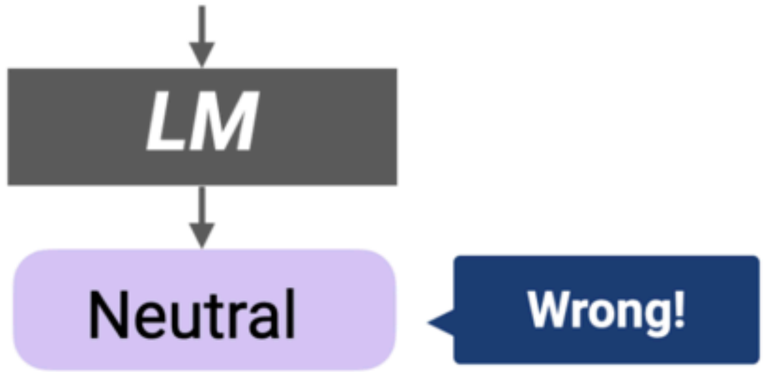
Colour-printed lithograph. Very good condition. \n Neutral

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

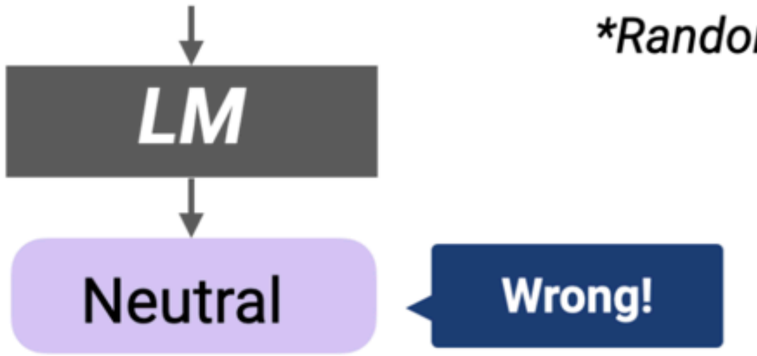
Circulation revenue has increased by 5% in Finland. \n Unanimity

Panostaja did not disclose the purchase price. \n Wave

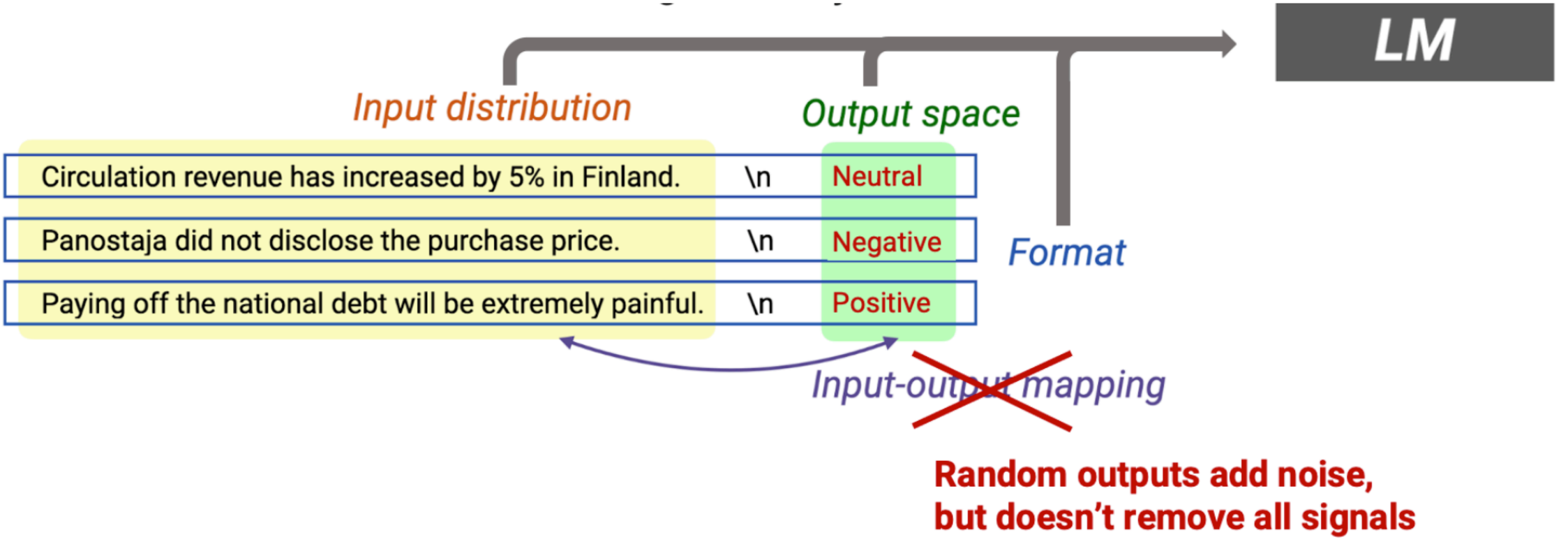
Paying off the national debt will be extremely painful. \n Guana

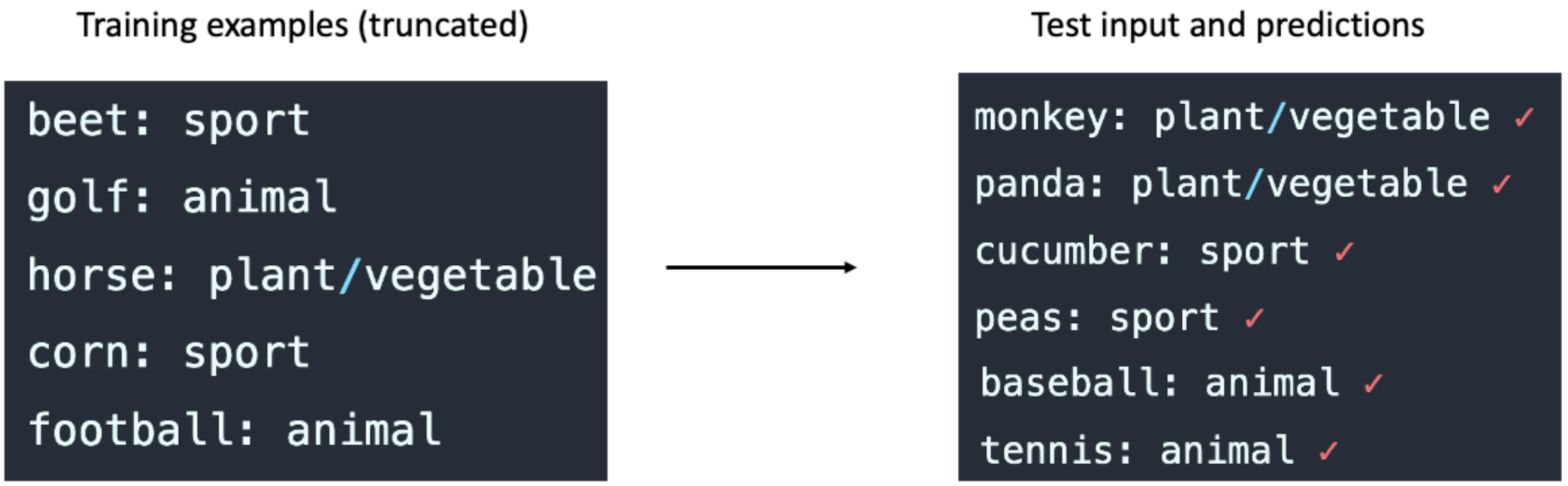
The company anticipated its operating profit to improve. \n _____

*Random English unigrams



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


LM
Input distribution
Output space
Format
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
Input-output mapping
Random outputs add noise,
but doesn't remove all signals



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