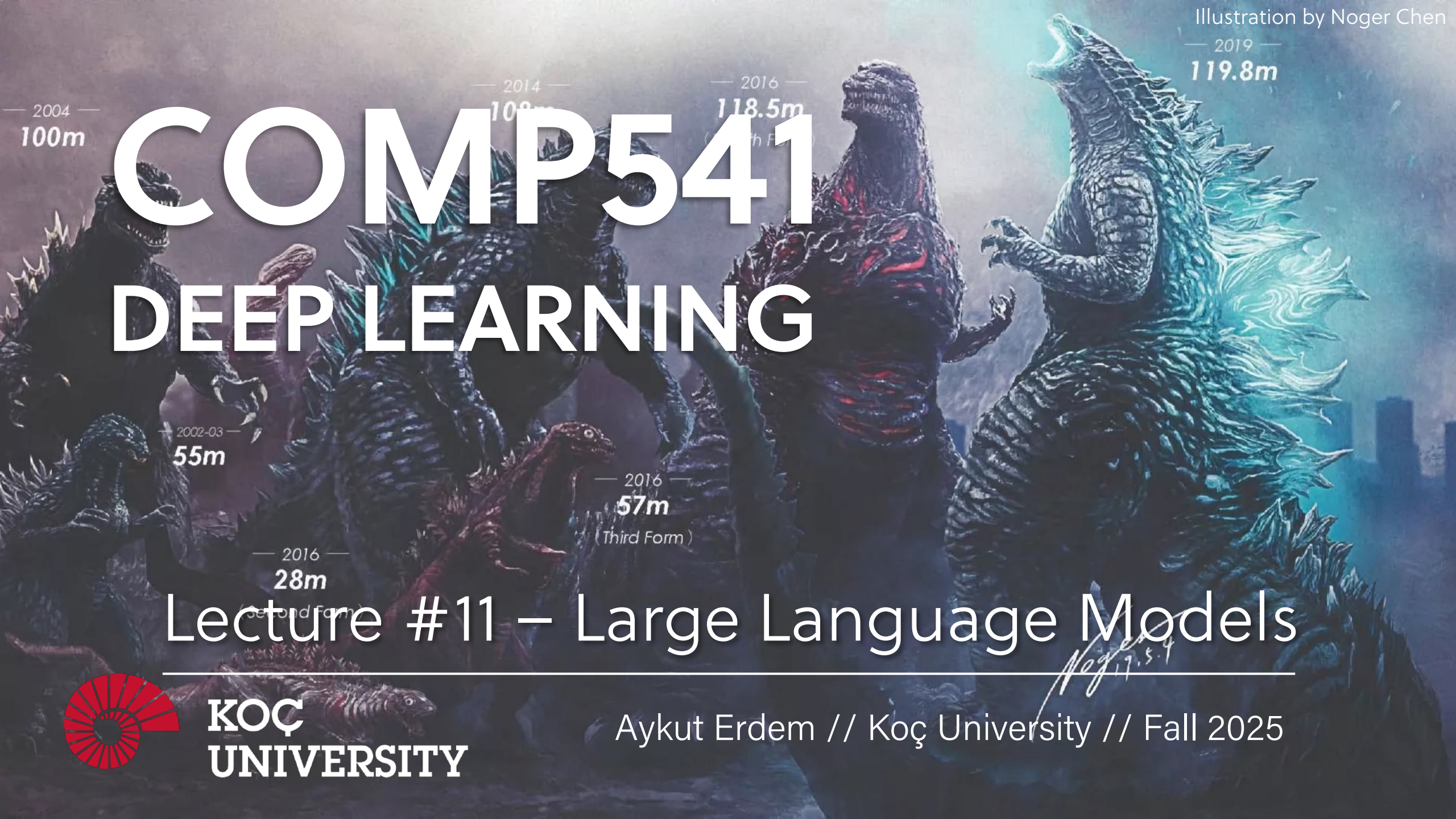


COMP541

DEEP LEARNING

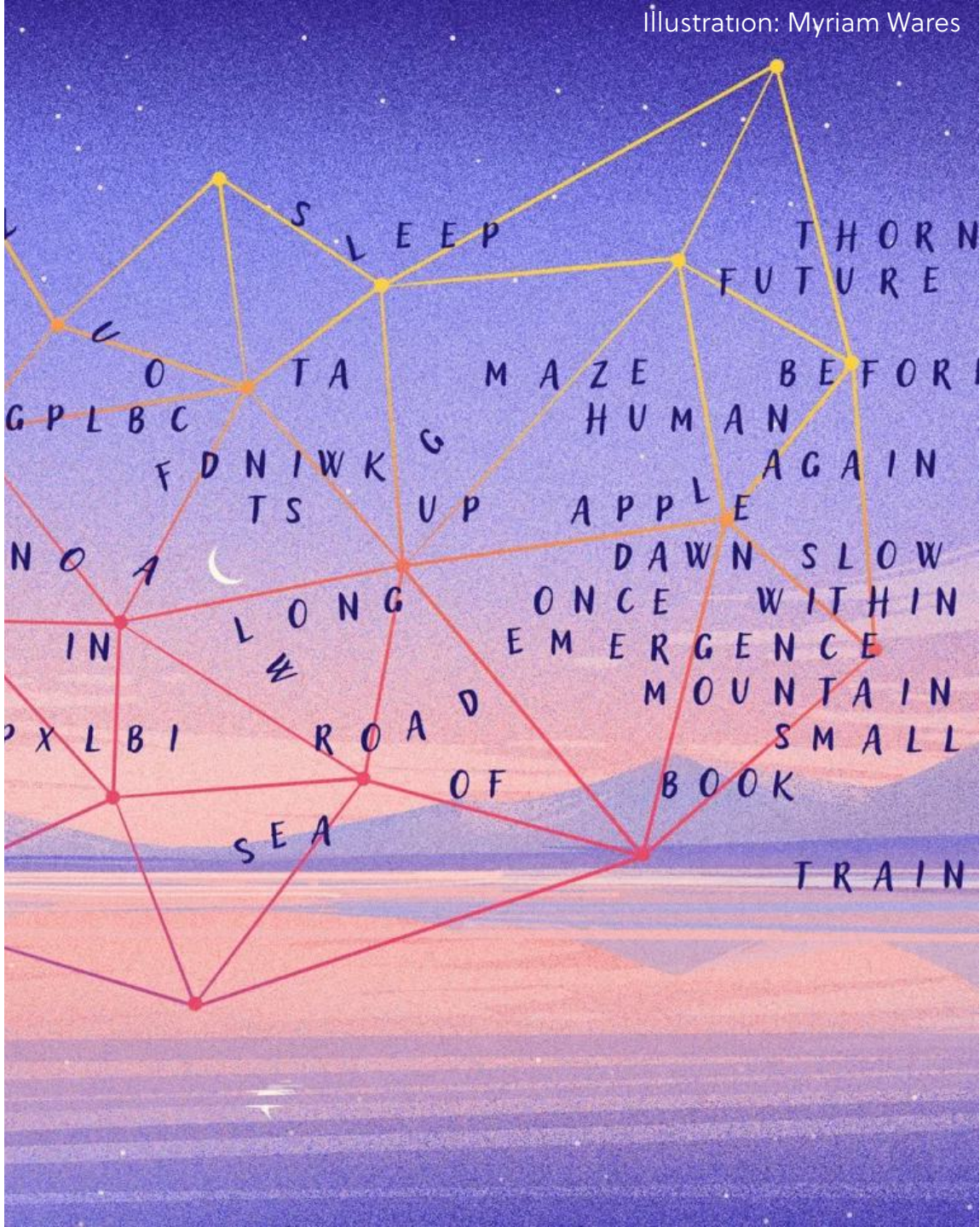
Lecture #11 – Large Language Models

KOÇ UNIVERSITY

Aykut Erdem // Koç University // Fall 2025

Previously on COMP541

- motivation and introduction
- introduction to language models
- history of neural language models
- pretrained language models



Lecture overview

- recap of language modeling
- GPT-3
- understanding in-context learning
- scaling laws
- Llama 3
- other LLMs
- long context models
- (open) model landscape in 2025

Disclaimer: Much of the material and slides for this lecture were borrowed from
— Danqi Chen and Sanjeev Arora's COS 597R class
— Graham Neubig's CS11-711 class
— Nathan Lambert's talk on Open Models in 2025

Recap of Language Modeling

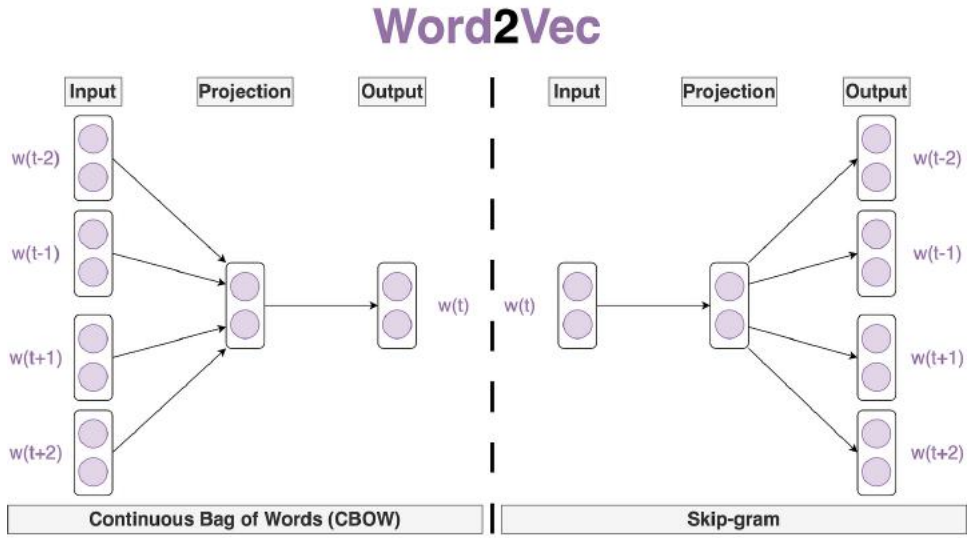
Word embeddings

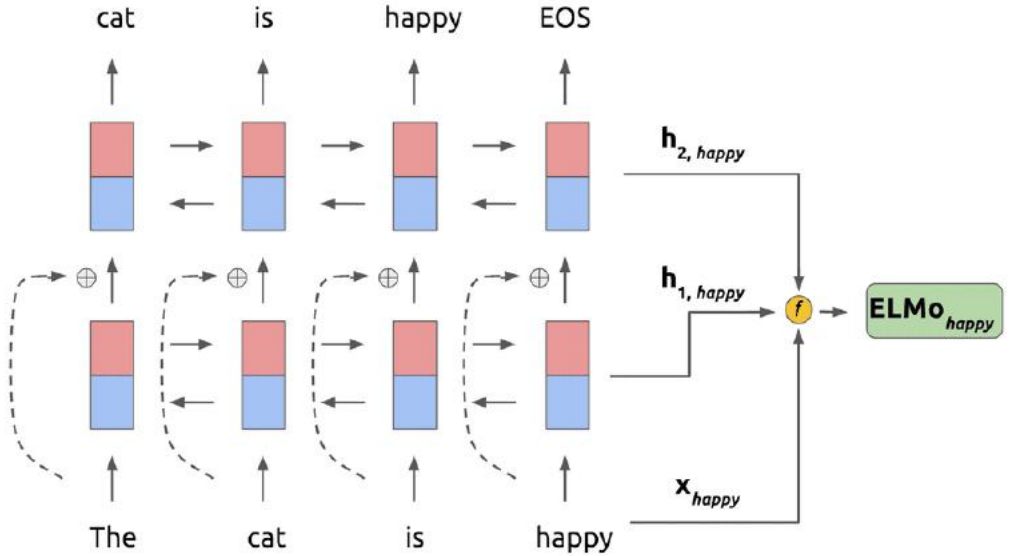
- **Word embeddings** e.g., word2vec (Mikolov et al.'13), GloVe (Pennington et al.'14)

  "single-layer representations were learned using word vectors"

- **Contextualized word embeddings** e.g., ELMo (Peters et al.'18), CoVe (McCann et al.'17)

  "RNNs with multiple layers of representations and contextual state were used to form stronger representations"



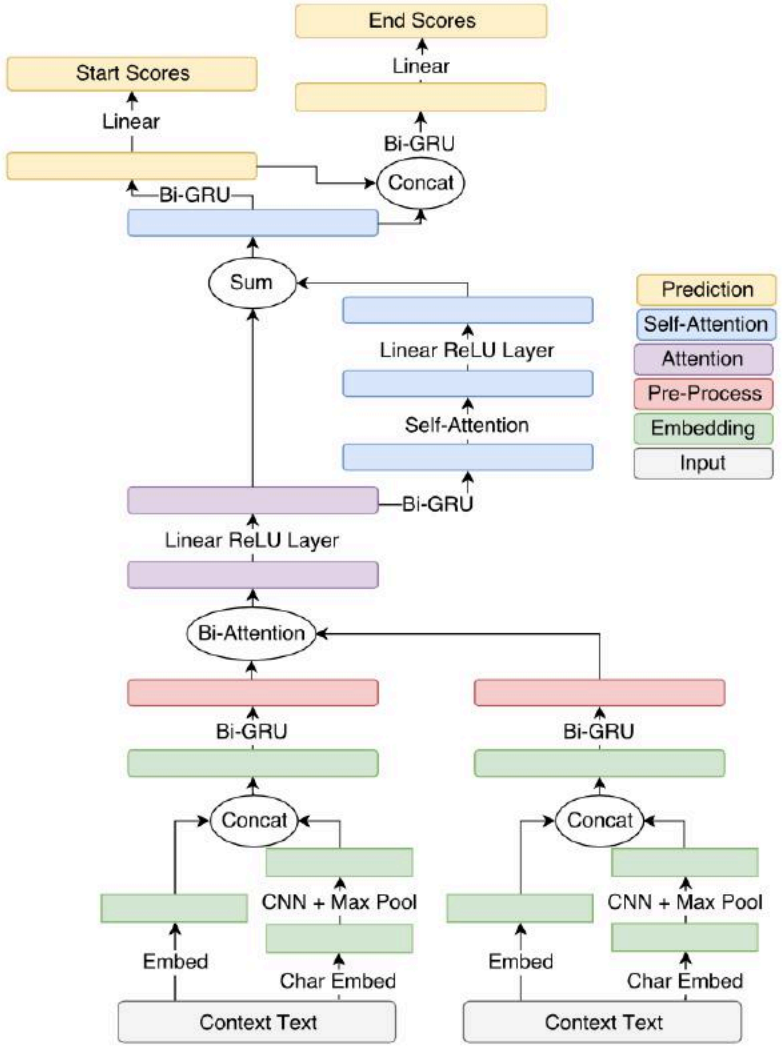


Used for task-specific neural architectures!

Word embeddings

- Word embeddings
- Contextualized word embeddings

Used for task-specific neural architectures!

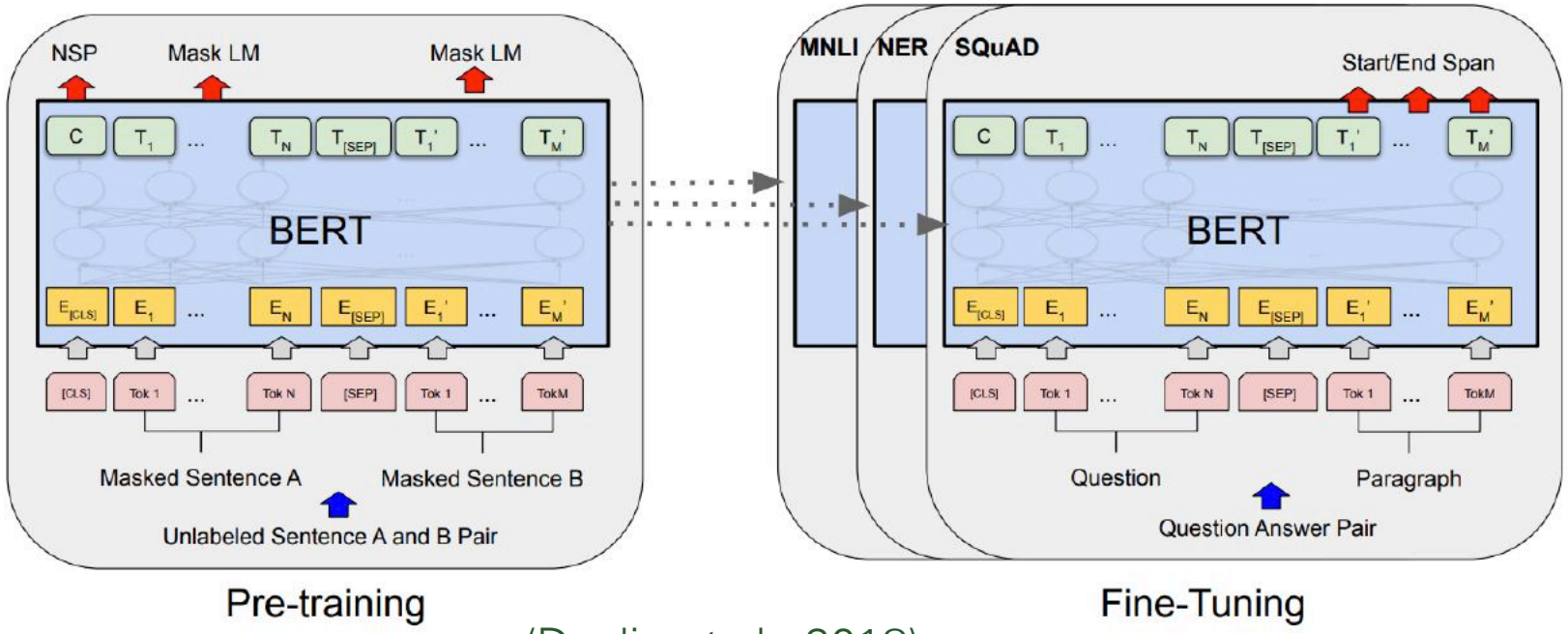


(Clark and Gardner, 2018)

One model for all tasks

- One pre-trained model for all tasks
  - BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019)
  - T5 (Raffel et al., 2019), BART (Lewis et al., 2019)
  - GPT-1 (Radford et al., 2018), GPT-2 (Radford et al., 2019)

minimal modifications to downstream tasks
still fine-tuning on 10^3– 10^5 downstream examples

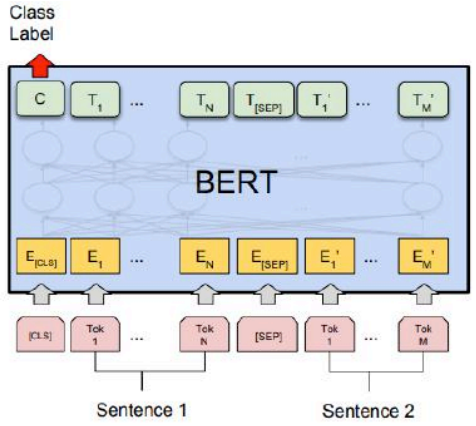


(Devlin et al., 2018)

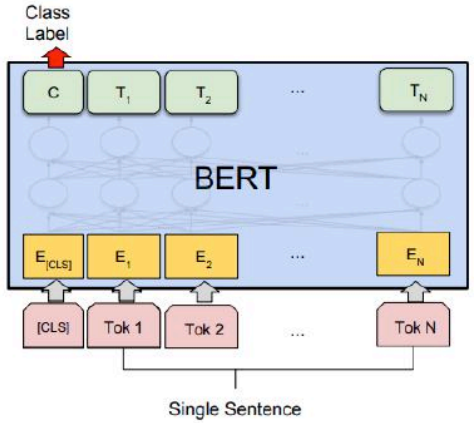
One model for all tasks

- One pre-trained model for all tasks

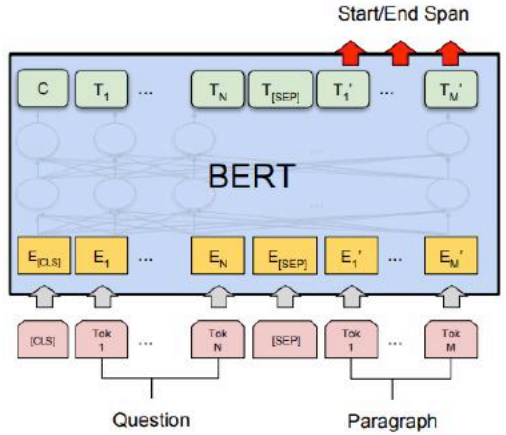
  minimal modifications to downstream tasks
  still fine-tuning on 10^3– 10^5 downstream examples



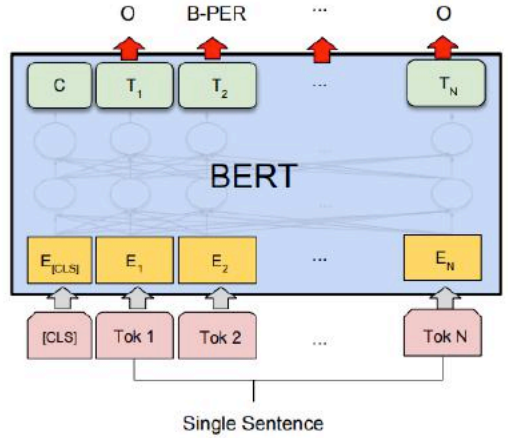
(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG



(b) Single Sentence Classification Tasks: SST-2, CoLA



(c) Question Answering Tasks:



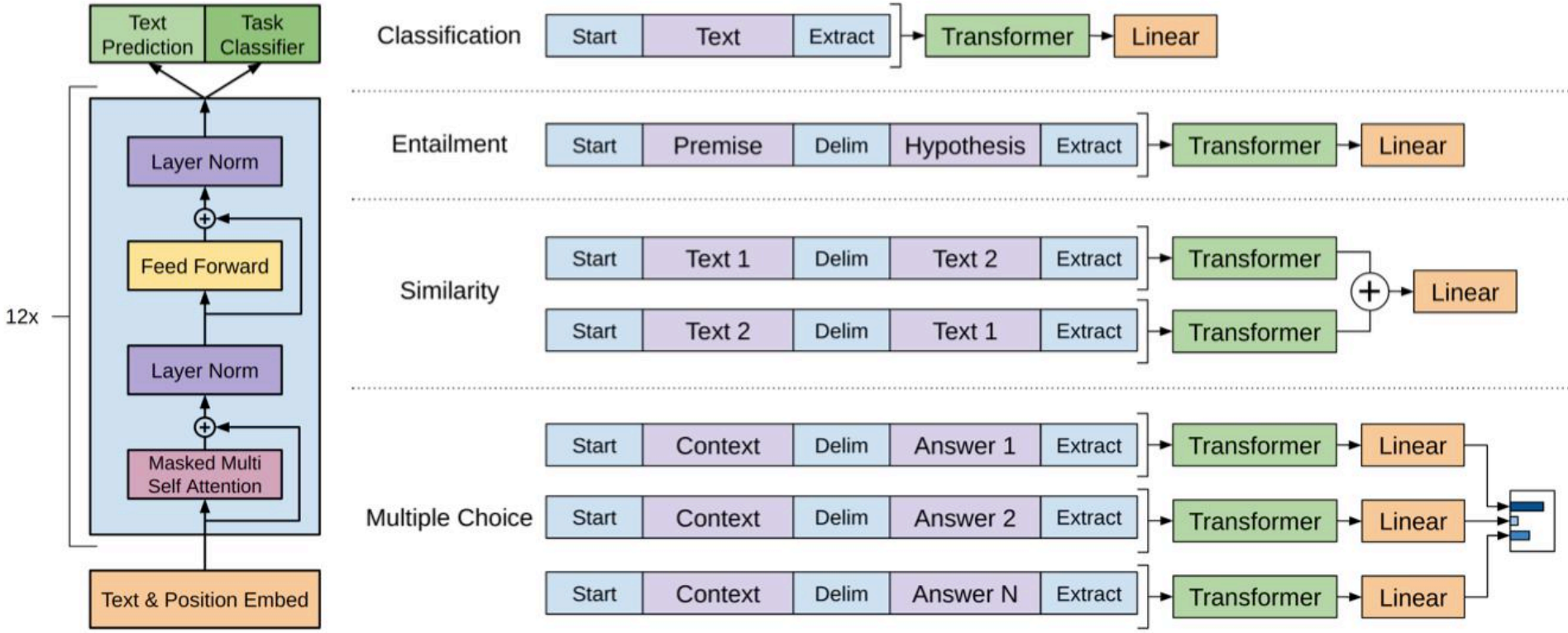
(d) Single Sentence Tagging Tasks:

(Devlin et al., 2018)

One model for all tasks

- One pre-trained model for all tasks

  minimal modifications to downstream tasks
  still fine-tuning on 10^3– 10^5 downstream examples



(Radford et al., 2018)

One model for all tasks

- **One pre-trained model for all tasks**
  - BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019) — encoder models
  - T5 (Raffel et al., 2019), BART (Lewis et al., 2019) — encoder-decoder models
  - GPT-1 (Radford et al., 2018), GPT-2 (Radford et al., 2019) — decoder models

- All based on **Transformers**
- They mainly differ in the pre-training objectives (slight difference in fine-tuning)
- Model sizes and pre-training data are also different!

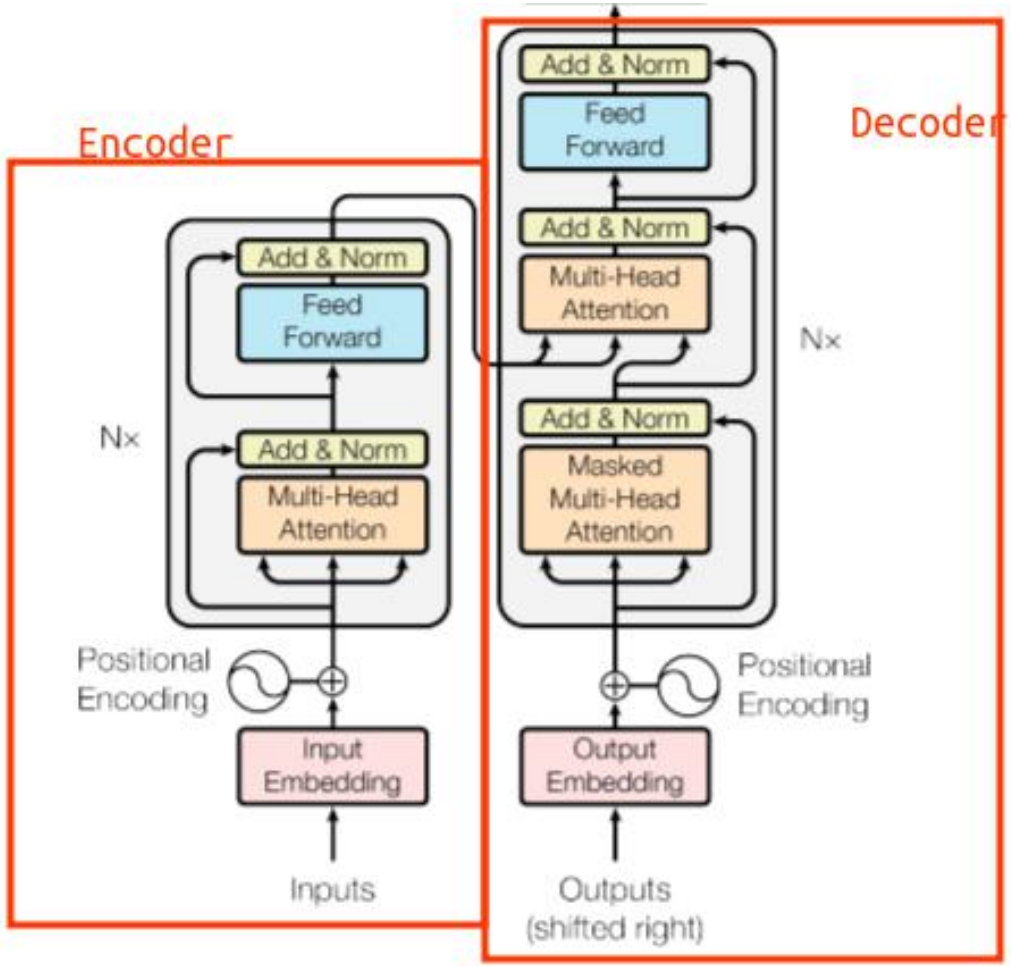
The Annotated Transformer

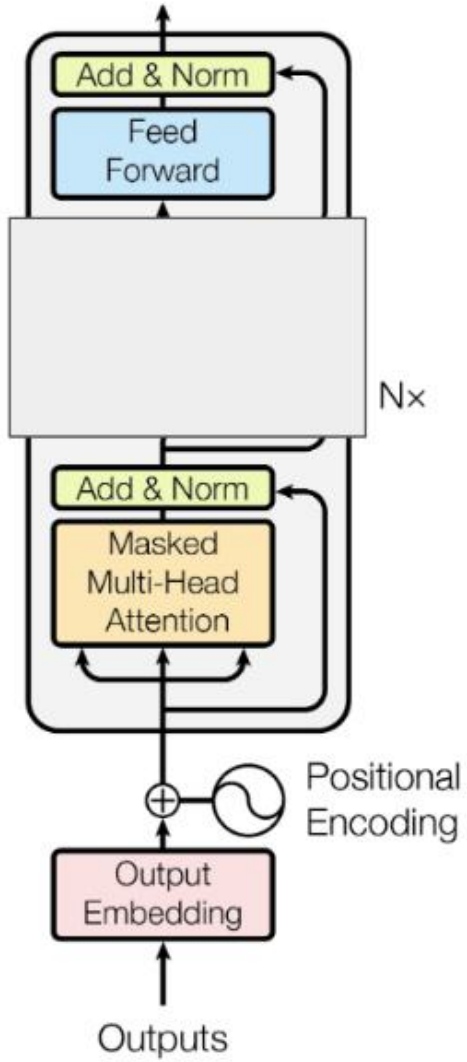
Attention is All You Need

Ashish Vaswani* Google Brain avaswani@google.com
Noam Shazeer* Google Brain noam@google.com
Niki Parmar* Google Research nikip@google.com
Jakob Uszkoreit* Google Research usz@google.com
Llion Jones* Google Research llion@google.com
Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu
Łukasz Kaiser* Google Brain lukaszkaiser@google.com
Illia Polosukhin* ‡ illia.polosukhin@gmail.com

- *v2022: Austin Huang, Suraj Subramanian, Jonathan Sum, Khalid Almubarak, and Stella Biderman.*
- *Original: Sasha Rush.*

One model for all tasks





Encoder vs. Decoder models

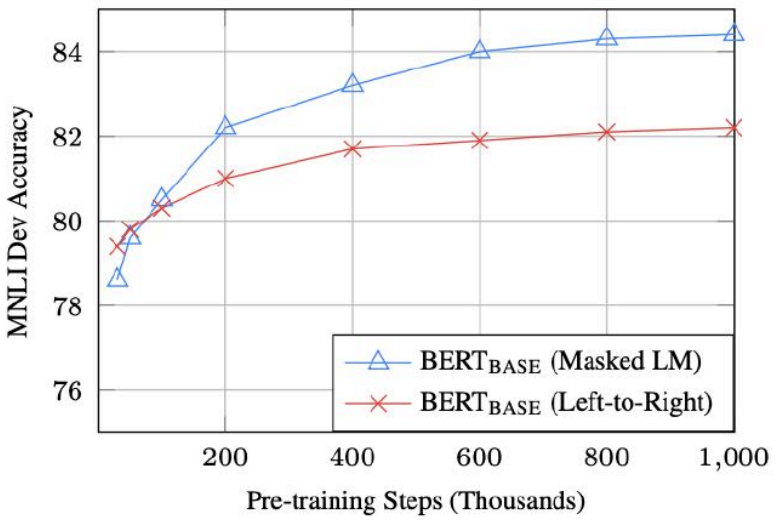


Figure 5: Ablation over number of training steps. This shows the MNLI accuracy after fine-tuning, starting from model parameters that have been pre-trained for k steps. The x-axis is the value of k.

(Devlin et al., 2018)

- BERT/RoBERTa: 110M/330M parameters
- T5: up to 11B parameters

Yi Tay

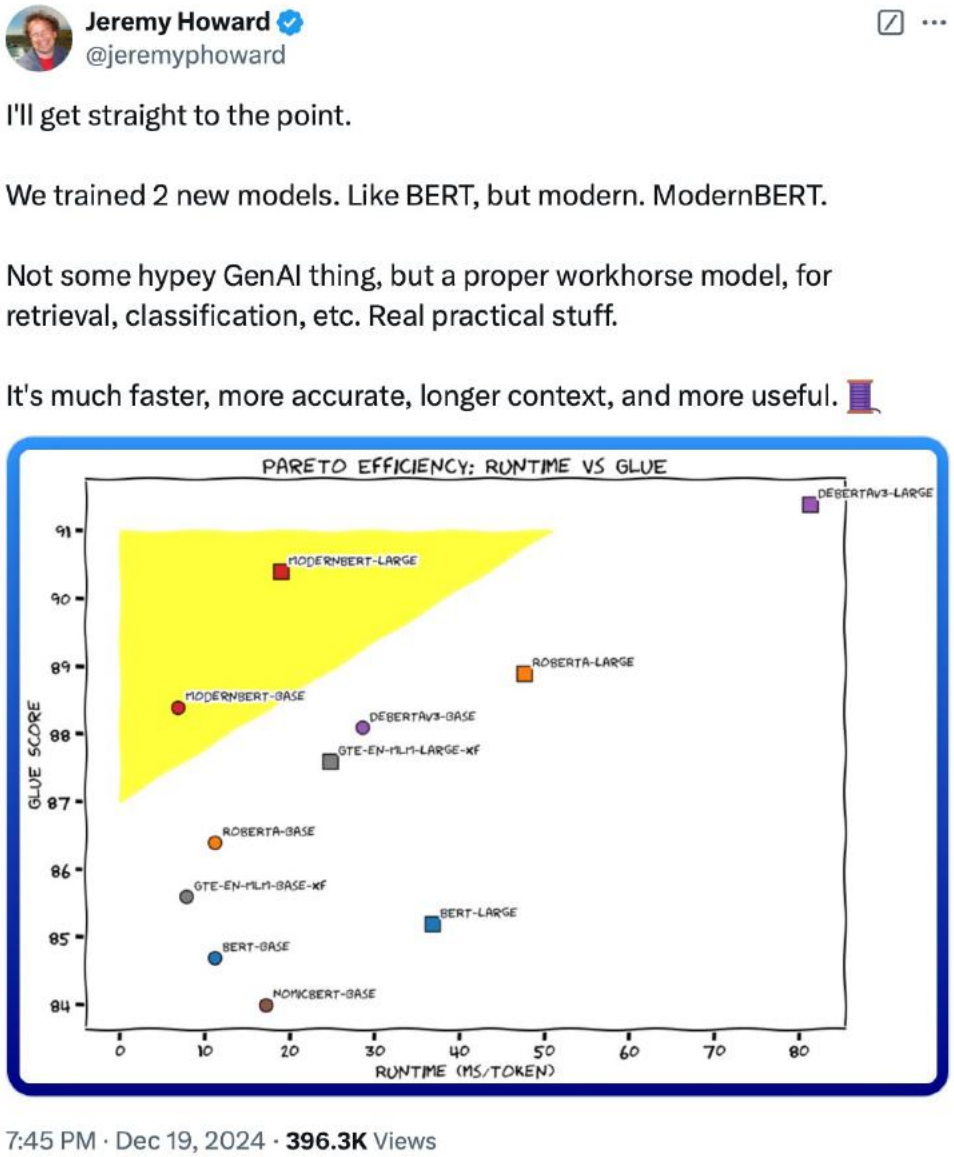
2024 JUL 16 - WRITTEN BY YI TAY

What happened to BERT & T5? On Transformer Encoders, PrefixLM and Denoising Objectives

https://www.yitay.net/blog/model-architecture-blogpost-encoders-prefixlm-denoising

- Encoder-only models can't generate text (easily); harder to scale up
- Bidirectional attention is only important at smaller scale?
- "Masking objectives" can be still combined with autoregressive LMs

Encoder vs. Decoder models



Smarter, Better, Faster, Longer: A Modern Bidirectional Encoder for Fast, Memory Efficient, and Long Context Finetuning and Inference

Benjamin Warner[1†] **Antoine Chaffin**[2†] **Benjamin Clavié**[1†]
Orion Weller[3] **Oskar Hallström**[2] **Said Taghadouini**[2]
Alexis Gallagher[1] **Raja Biswas**[1] **Faisal Ladhak**[4*] **Tom Aarsen**[5]
Nathan Cooper[1] **Griffin Adams**[1] **Jeremy Howard**[1] **Iacopo Poli**[2]

[1]Answer.AI [2]LightOn [3]Johns Hopkins University [4]NVIDIA [5]HuggingFace

*†: core authors, *: work done while at Answer.AI*

Correspondence: {bw,bc}@answer.ai, antoine.chaffin@lighton.ai

Abstract

Encoder-only transformer models such as BERT offer a great performance-size tradeoff for retrieval and classification tasks with respect to larger decoder-only models. Despite being the workhorse of numerous production pipelines, there have been limited Pareto improvements to BERT since its release. In this paper, we introduce ModernBERT, bringing modern model optimizations to encoder-only models and representing a major Pareto improvement over older encoders. Trained on 2 trillion tokens with a native 8192 sequence length, ModernBERT models exhibit state-of-the-art results on a large pool of evaluations encompassing diverse classification tasks and both single and multi-vector retrieval on different domains (including code). In addition to strong downstream performance, ModernBERT is also the most speed and memory efficient encoder and is designed for inference on common GPUs.

1 Introduction

After the release of BERT (Devlin et al., 2019), encoder-only transformer-based (Vaswani et al., 2017) language models dominated most applications of modern Natural Language Processing (NLP). Despite the rising popularity of Large Language Models (LLMs) such as GPT (Radford et al., 2018, 2019; Brown et al., 2020), Llama (Touvron

option against encoder-decoder and decoder-only language models when dealing with substantial amounts of data (Penedo et al., 2024).

Encoder models are particularly popular in Information Retrieval (IR) applications, e.g., semantic search, with notable progress on leveraging encoders for this task (Karpukhin et al., 2020; Khattab and Zaharia, 2020). While LLMs have taken the spotlight in recent years, they have also motivated a renewed interest in encoder-only models for IR. Indeed, encoder-based semantic search is a core component of Retrieval-Augmented Generation (RAG) pipelines (Lewis et al., 2020), where encoder models are used to retrieve and feed LLMs with context relevant to user queries.

Encoder-only models are also still frequently used for a variety of discriminative tasks such as classification (Tunstall et al., 2022) or Natural Entity Recognition (NER) (Zaratiana et al., 2024), where they often match the performance of specialized LLMs. Here again, they can be used in conjunction with LLMs, for example detecting toxic prompts (Ji et al., 2023; Jiang et al., 2024b) and preventing responses, or routing queries in an agentic framework (Yao et al., 2023; Schick et al., 2023).

Surprisingly, these pipelines currently rely on older models, and quite often on the original BERT itself as their backbone (Wang et al., 2022; Xiao et al., 2023), without leveraging improvements de-

Recap: Probabilistic Language Models

$$P(X)$$

Sentence/Document

A generative model that calculates
the probability of language

Recap: Auto-regressive Language Models

$$P(X) = \prod_{i=1}^{I} P(x_i \mid x_1, \ldots, x_{i-1})$$

Next Token Context

Recap: Next Token Prediction

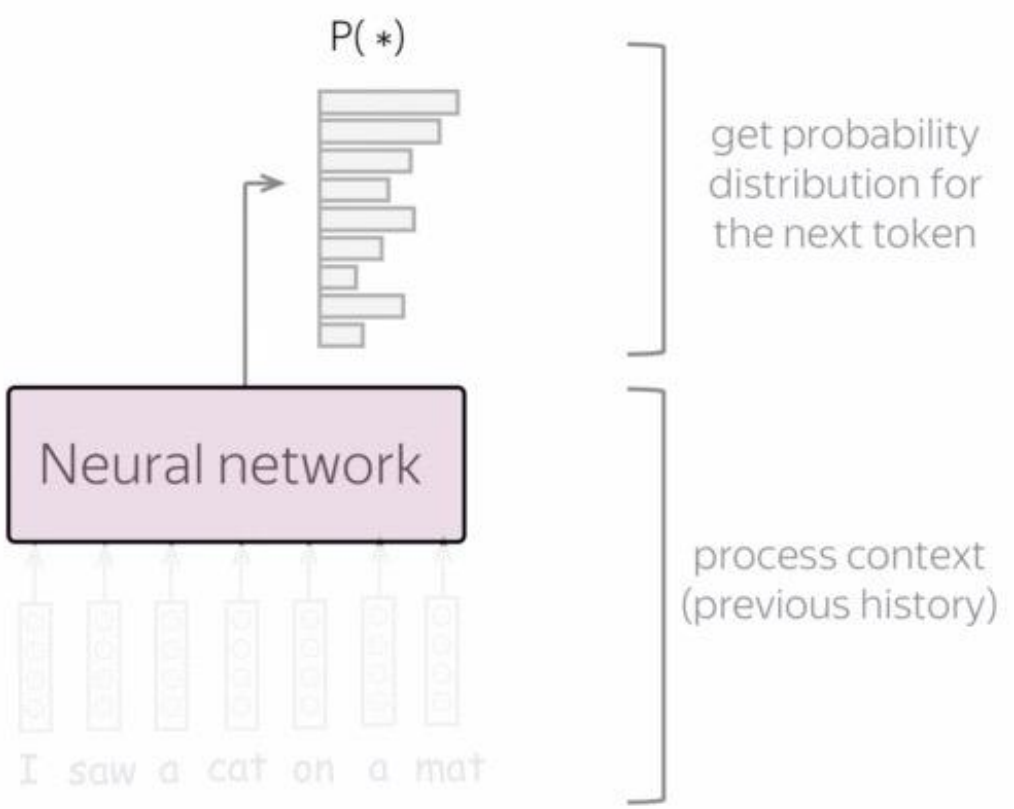


Image Credit: https://lena-voita.github.io/nlp_course/language_modeling.html

- This is classification! We can think of neural language models as neural classifiers. They classify prefix of a text into |V| classes, where the classes are vocabulary tokens.

Recap: Next Token Prediction

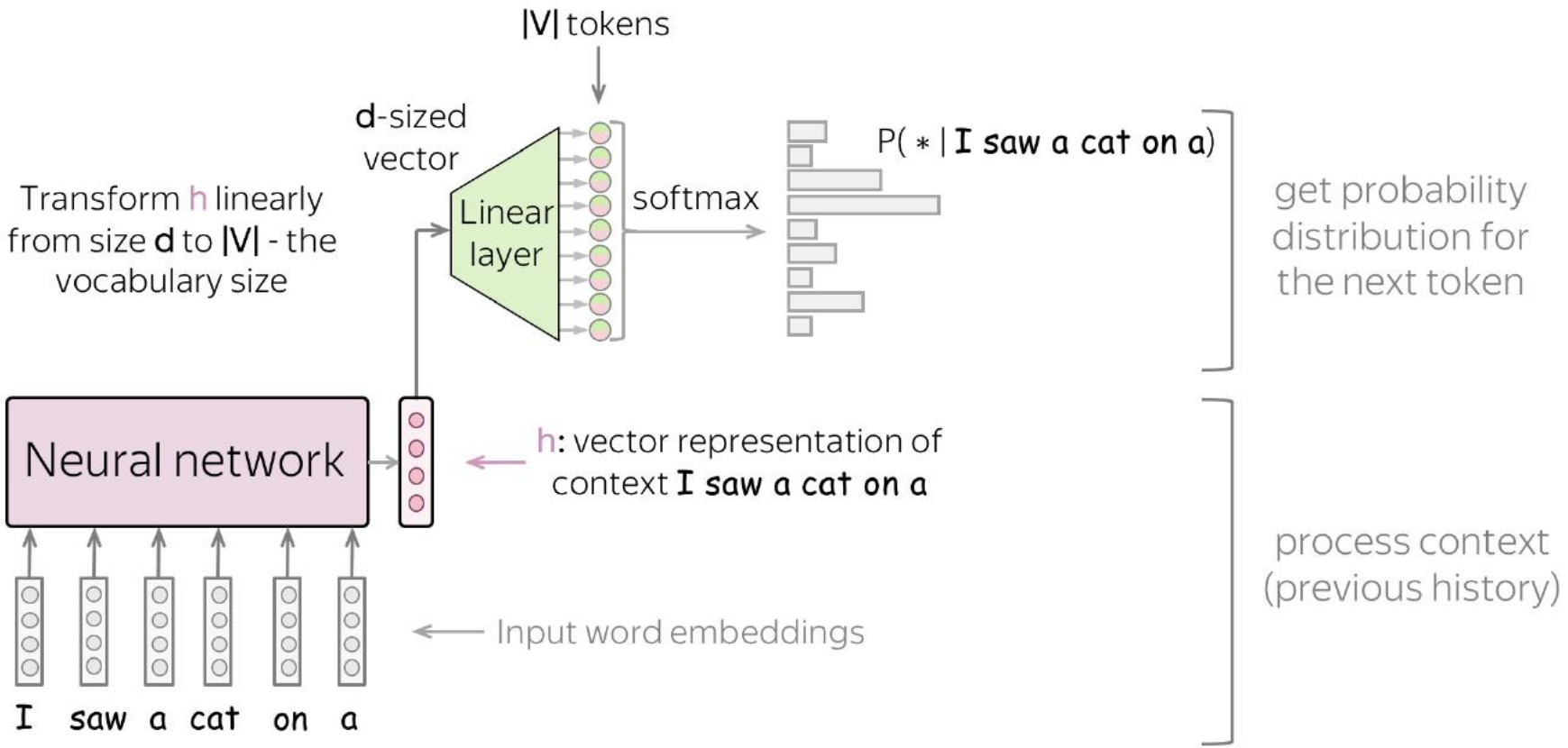
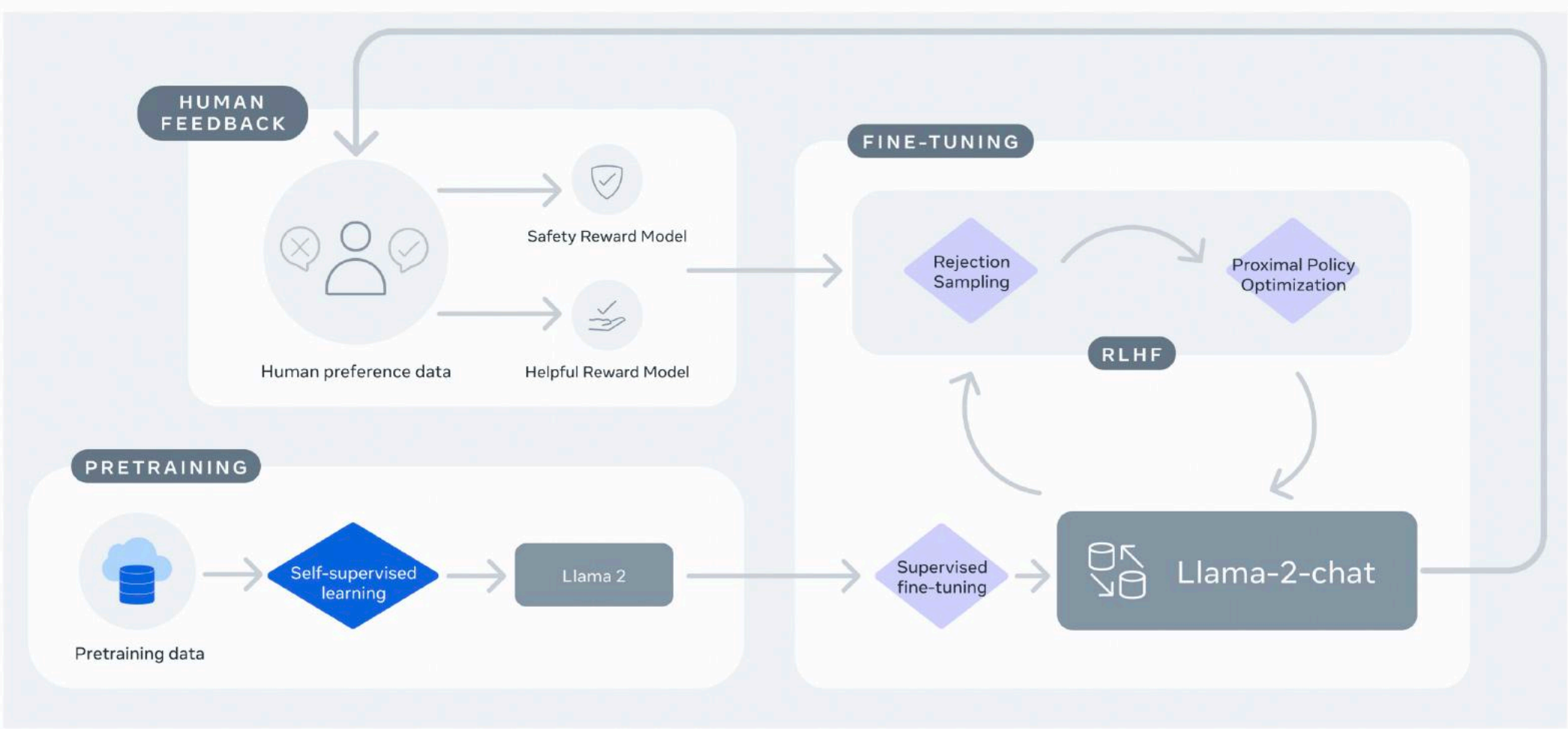


Image Credit: https://lena-voita.github.io/nlp_course/language_modeling.html

- feed word embedding for previous (context) words into a network;
- get vector representation of context from the network;
- from this vector representation, predict a probability distribution for the next token.

Overview of LLMs Training



Pre-training → Supervised Fine-tuning (SFT) → RLHF
Pre-training → Post-training
Pre-training → Mid-training → Post-training

More next week!

GPT-3

GPT-3

arXiv
https://arxiv.org › cs

[2005.14165] Language Models are Few-Shot Learners

by TB Brown · 2020 · Cited by 31178 — Specifically, we train **GPT-3**, an autoregressive language model with 175 billion parameters, 10x more than any previous non-sparse languag…

[Submitted on 28 May 2020 (v1), last revised 22 Jul 2020 (this version, v4)]

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

Recent work has demonstrated substantial gains on many NLP tasks and benchmarks by pre-training on a large corpus of text followed by fine-tuning on a specific task. While typically task-agnostic in architecture, this method still requires task-specific fine-tuning datasets of thousands or tens of thousands of examples. By contrast, humans can generally perform a new language task from only a few examples or from simple instructions - something which current NLP systems still largely struggle to do. Here we show that scaling up language models greatly improves task-agnostic, few-shot performance, sometimes even reaching competitiveness with prior state-of-the-art fine-tuning approaches. Specifically, we train GPT-3, an autoregressive language model with 175 billion parameters, 10x more than any previous non-sparse language model, and test its performance in the few-shot setting. For all tasks, GPT-3 is applied without any gradient updates or fine-tuning, with tasks and few-shot demonstrations specified purely via text interaction with the model. GPT-3 achieves strong performance on many NLP datasets, including translation, question-answering, and cloze tasks, as well as several tasks that require on-the-fly reasoning or domain adaptation, such as unscrambling words, using a novel word in a sentence, or performing 3-digit arithmetic. At the same time, we also identify some datasets where GPT-3's few-shot learning still struggles, as well as some datasets where GPT-3 faces methodological issues related to training on large web corpora. Finally, we find that GPT-3 can generate samples of news articles which human evaluators have difficulty distinguishing from articles written by humans. We discuss broader societal impacts of this finding and of GPT-3 in general.

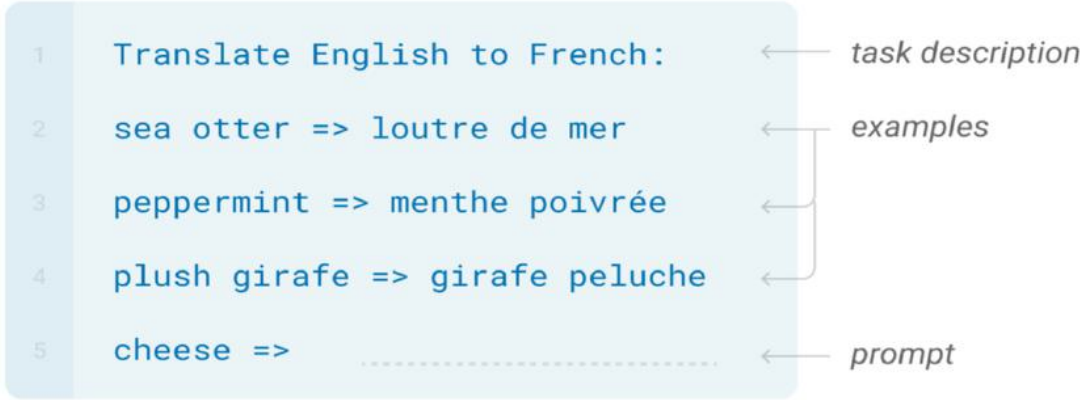
"Language models form the backbone of modern techniques for solving a range of problems in natural language processing. The paper shows that when such language models are scaled up to an unprecedented number of parameters, **the language model itself can be used as a few-shot learner that achieves very competitive performance on many of these problems without any additional training**. This is a very surprising result that is expected to have substantial impact in the field, and that is likely to withstand the test of time. In addition to the scientific contribution of the work, the paper also **presents a very extensive and thoughtful exposition of the broader impact of the work**, which may serve as an example to the NeurIPS community on how to think about the real-world impact of the research performed by the community."

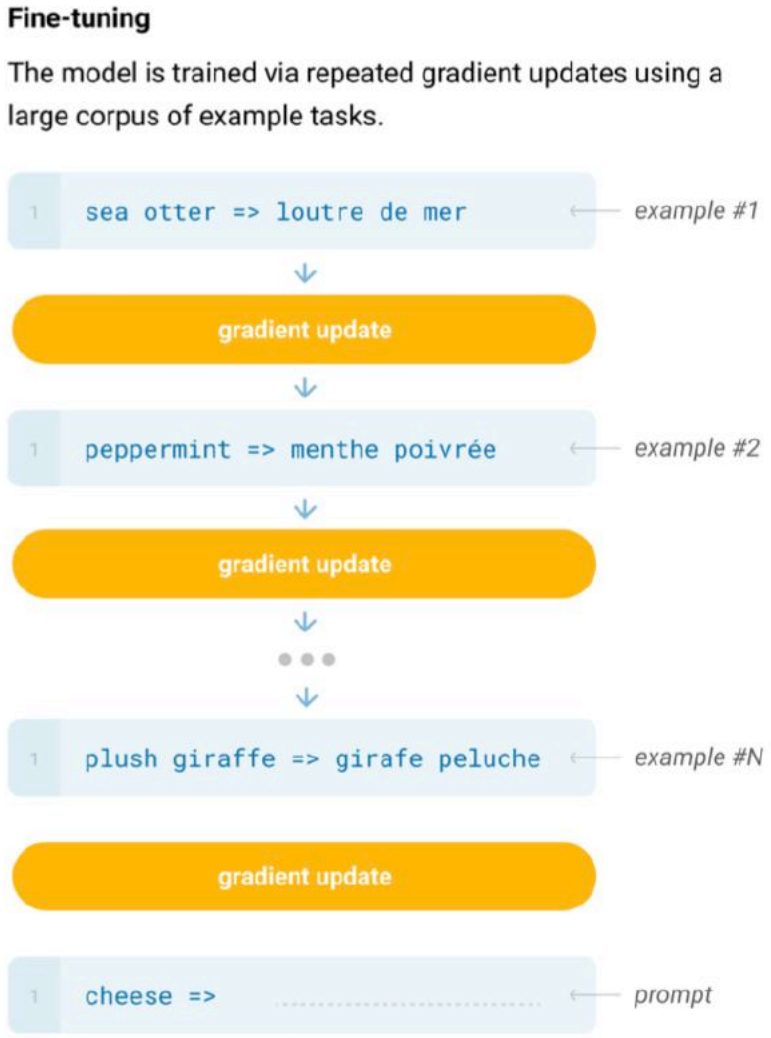
https://neuripsconf.medium.com/announcing-the-neurips-2020-award-recipients-73e4d3101537

GPT-3: main contributions

- An autoregressive language model of 175B parameters, 10x larger than any previous LMs
- Introduced the concept of "in-context learning", and showed competitive performance

In-context learning: you can perform a task from only **a few examples** or **simple instructions** without any gradient updates or fine-tuning!

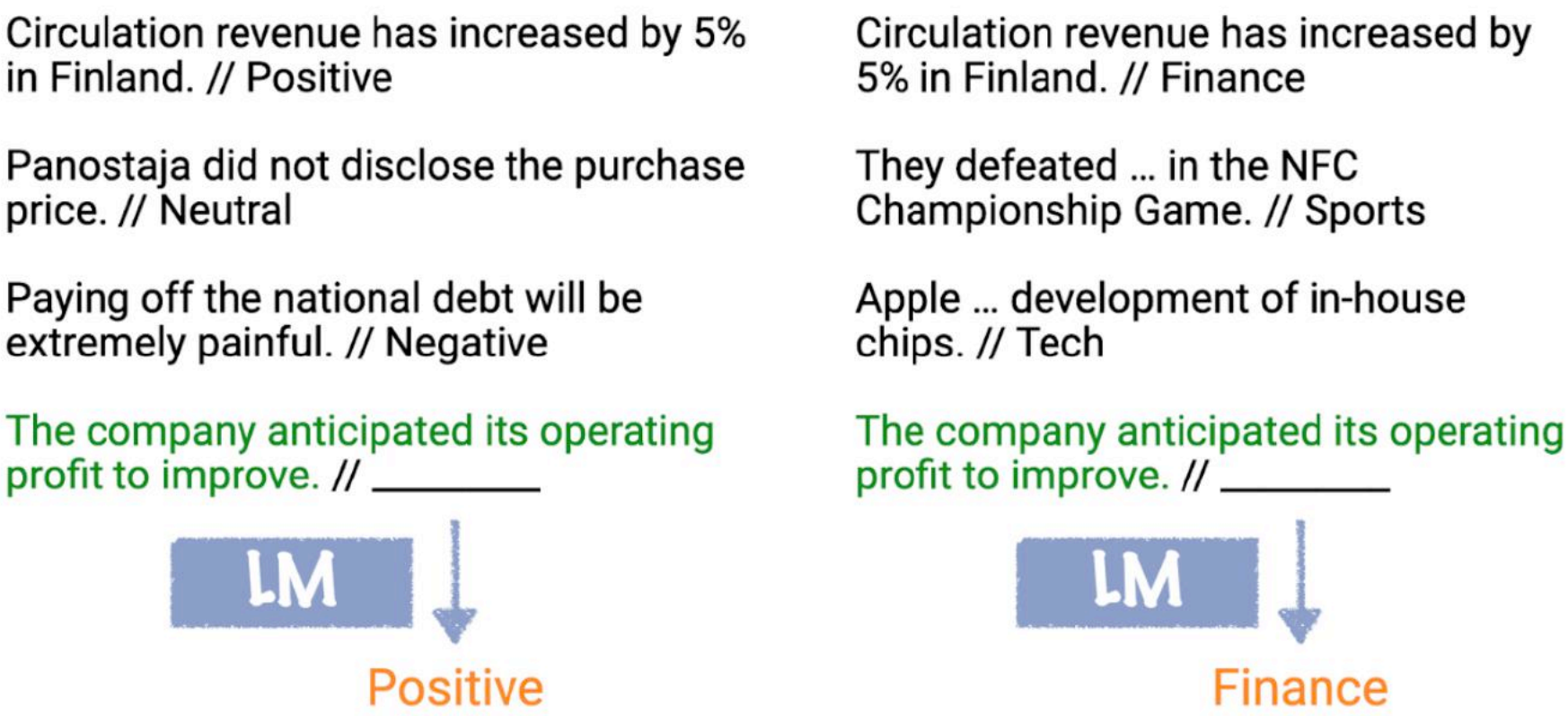




https://ai.stanford.edu/blog/understanding-incontext/

GPT-3: main contributions

In-context learning: you can perform a task from only **a few examples** or **simple instructions** without any gradient updates or fine-tuning!



https://ai.stanford.edu/blog/understanding-incontext/

GPT-3: main contributions

In-context learning: you can perform a task from only **a few examples** or **simple instructions** without any gradient updates or fine-tuning!

- **Interesting note:** the idea of in-context learning starts from GPT-2, "though with much more limited results and no systematic study."

3.7. Translation

We test whether GPT-2 has begun to learn how to translate from one language to another. In order to help it infer that this is the desired task, we condition the language model on a context of example pairs of the format `english sentence = french sentence` and then after a final prompt of `english sentence =` we sample from the model with greedy decoding and use the first generated sentence as the translation. On the WMT-14 English-French

3.8. Question Answering

tively. Similar to translation, the context of the language model is seeded with example question answer pairs which helps the model infer the short answer style of the dataset. GPT-2 answers 4.1% of questions correctly when evaluated by the exact match metric commonly used on reading

(Radford et al., 2019)

Why few-shot learning?

- Collecting large supervised training sets is expensive

Corpus	|Train|	|Test|	Task	Metrics	Domain
			Single-Sentence Tasks		
CoLA	8.5k	**1k**	acceptability	Matthews corr.	misc.
SST-2	67k	1.8k	sentiment	acc.	movie reviews
			Similarity and Paraphrase Tasks		
MRPC	3.7k	1.7k	paraphrase	acc./F1	news
STS-B	7k	1.4k	sentence similarity	Pearson/Spearman corr.	misc.
QQP	364k	**391k**	paraphrase	acc./F1	social QA questions
			Inference Tasks		
MNLI	393k	**20k**	NLI	matched acc./mismatched acc.	misc.
QNLI	105k	5.4k	QA/NLI	acc.	Wikipedia
RTE	2.5k	3k	NLI	acc.	news, Wikipedia
WNLI	634	**146**	coreference/NLI	acc.	fiction books

GLUE (Devlin et al., 2018)

Why few-shot learning?

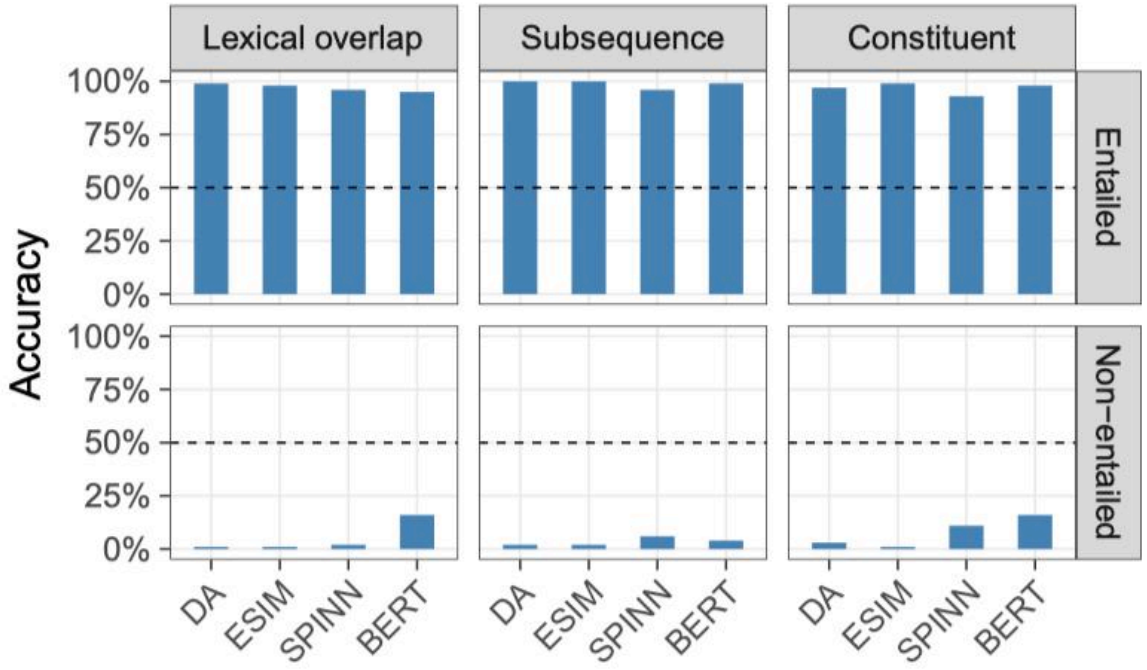
- Fine-tuning can exploit **spurious correlation** and do not generalize well out-of-distribution

NATURAL LANGUAGE INFERENCE E.G., MNLI

- **Premise:** The banker near the judge saw the actor.
- **Hypothesis:** The banker saw the actor.
- **Label:** Entailment

Lexical overlap heuristic: a premise entails all hypotheses constructed from words in the premise

- **Premise:** The doctors visited the lawyer.
- **Hypothesis:** The lawyer visited the doctors.
- **Label:** Not Entailment ❌



(McCoy et al., 2019)

Why few-shot learning?

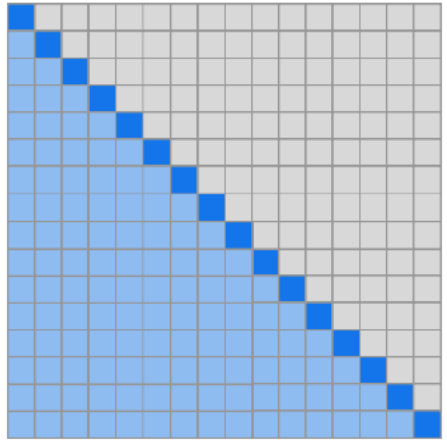
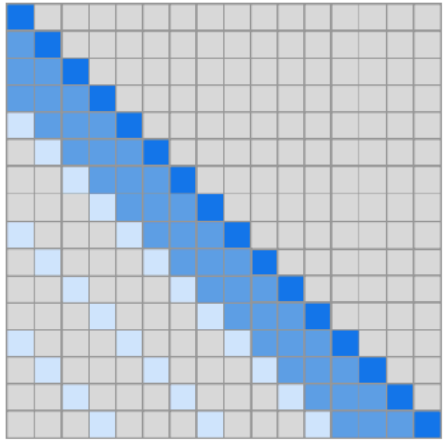
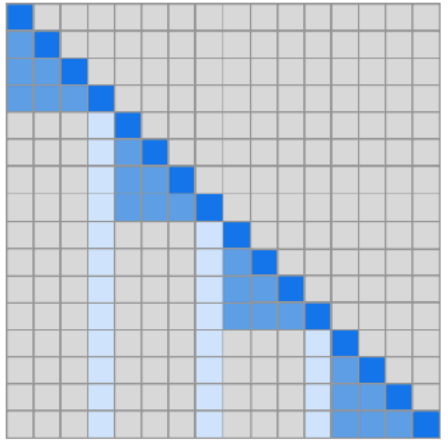
- Humans do not require large supervised datasets to learn most language tasks
- It allows humans to seamlessly **mix together** or **switch** between many tasks and tasks when interacting with NLP systems
  - Fluidity
  - Generality

Overview of GPT-3

- GPT-3 is a Transformer decoder only trained on large amounts of unlabeled text
- Training objective: next-token prediction

$$L_1(\mathcal{U}) = \sum_i \log P\left(u_i \mid u_{i-k}, \ldots, u_{i-1}; \Theta\right)$$

- Model architecture the same as GPT-2, including modified initialization, pre-normalization
  - Except that "we use alternating dense and locally banded sparse attention patterns in the layers of the Transformer"

(Child et al., 2019)

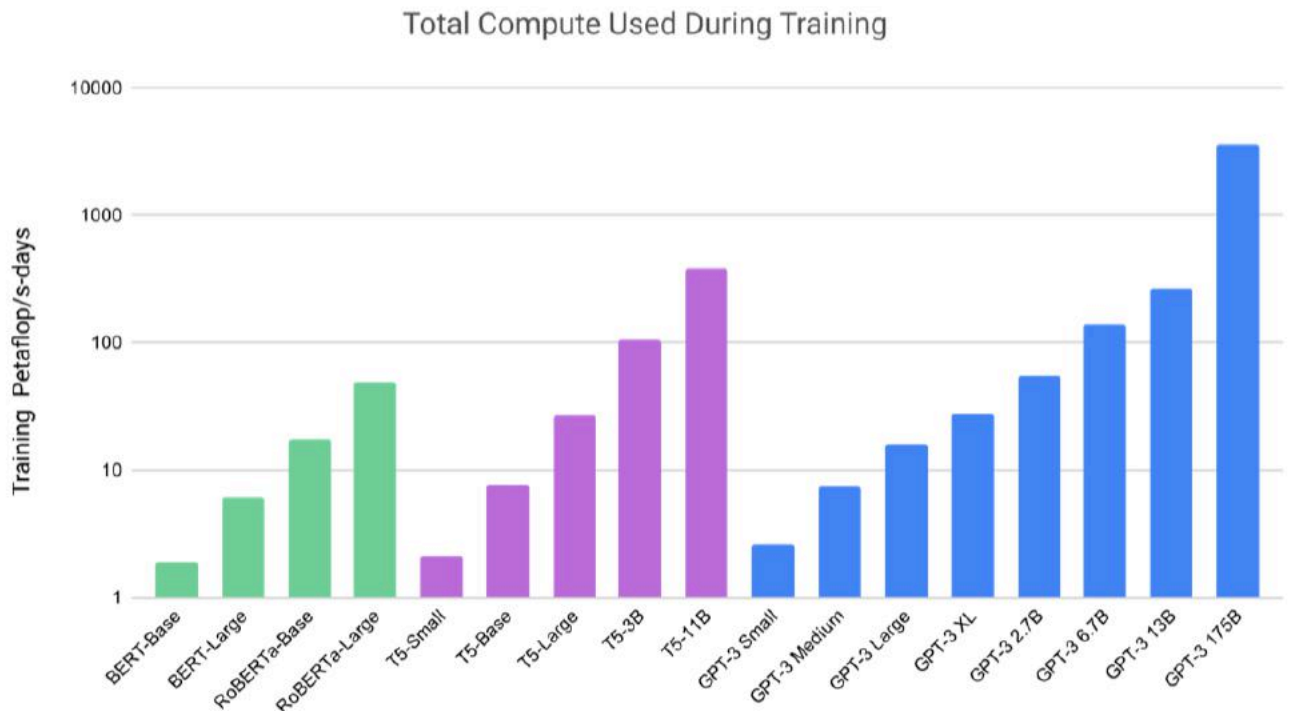
Overview of GPT-3

- GPT-3 is a Transformer decoder only trained on large amounts of unlabeled text
- All models were trained on 300B tokens

Model Name	n_{params}	n_{layers}	d_{model}	n_{heads}	d_{head}	Batch Size	Learning Rate
GPT-3 Small	125M	12	768	12	64	0.5M	6.0×10^{-4}
GPT-3 Medium	350M	24	1024	16	64	0.5M	3.0×10^{-4}
GPT-3 Large	760M	24	1536	16	96	0.5M	2.5×10^{-4}
GPT-3 XL	1.3B	24	2048	24	128	1M	2.0×10^{-4}
GPT-3 2.7B	2.7B	32	2560	32	80	1M	1.6×10^{-4}
GPT-3 6.7B	6.7B	32	4096	32	128	2M	1.2×10^{-4}
GPT-3 13B	13.0B	40	5140	40	128	2M	1.0×10^{-4}
GPT-3 175B or "GPT-3"	175.0B	96	12288	96	128	3.2M	0.6×10^{-4}

- **Scaling laws:** "scaling of validation loss should be approximately a smooth power law as a function of size"
- Larger models typically use a larger batch size but require a smaller learning rate
- **Context window size = 2048**
- Use a lot of "model parallelism" during training
- Use Adam optimizer $\beta_1 = 0.9$, $\beta_2 = 0.95$, and $\epsilon = 10^{-8}$

GPT-3: Training compute



Model	Total train compute (PF-days)	Total train compute (flops)	Params (M)	Training tokens (billions)
T5-Small	2.08E+00	1.80E+20	60	1,000
T5-Base	7.64E+00	6.60E+20	220	1,000
T5-Large	2.67E+01	2.31E+21	770	1,000
T5-3B	1.04E+02	9.00E+21	3,000	1,000
T5-11B	3.82E+02	3.30E+22	11,000	1,000
BERT-Base	1.89E+00	1.64E+20	109	250
BERT-Large	6.16E+00	5.33E+20	355	250
RoBERTa-Base	1.74E+01	1.50E+21	125	2,000
RoBERTa-Large	4.93E+01	4.26E+21	355	2,000
GPT-3 Small	2.60E+00	2.25E+20	125	300
GPT-3 Medium	7.42E+00	6.41E+20	356	300
GPT-3 Large	1.58E+01	1.37E+21	760	300
GPT-3 XL	2.75E+01	2.38E+21	1,320	300
GPT-3 2.7B	5.52E+01	4.77E+21	2,650	300
GPT-3 6.7B	1.39E+02	1.20E+22	6,660	300
GPT-3 13B	2.68E+02	2.31E+22	12,850	300
GPT-3 175B	3.64E+03	3.14E+23	174,600	300

"We train much larger models on many fewer tokens"

GPT-3: Training data

- Common Crawl (CC) + a set of high-quality, curated data
  - Common Crawl is a nonprofit organization that crawls the web and freely provides its archives and datasets to the public.
  - Lots of low-quality and duplicated content - requires heavy filtering
  - We will see lots of efforts later, e.g., RefineWeb, FineWeb-edu
  - Data in the mix: WebText, Books1, Books2, English Wikipedia
- Filtering CC:
  - Filtering based on similarity to a range of high-quality reference corpora
  - Fuzzy deduplication at the document level
- Data sampling: sample from high-quality data more frequently!

GPT-3: Training data

Dataset	Quantity (tokens)	Weight in training mix	Epochs elapsed when training for 300B tokens
Common Crawl (filtered)	410 billion	60%	0.44
WebText2	19 billion	22%	2.9
Books1	12 billion	8%	1.9
Books2	55 billion	8%	0.43
Wikipedia	3 billion	3%	3.4

Approach

- **Few-shot:** a few demonstrations are prepended in the context (no weights updated allowed)
  - The demonstrations are randomly sampled from training set
  - K: typically 10-100, depending on how many examples can fit in context (2048)
  - Not always "the larger K, the better" => use a development set to decide K
  - Optionally add a natural language prompt
- **One-shot:** special case when K = 1.

  "it most closely matches the way in which some tasks are communicated to humans"

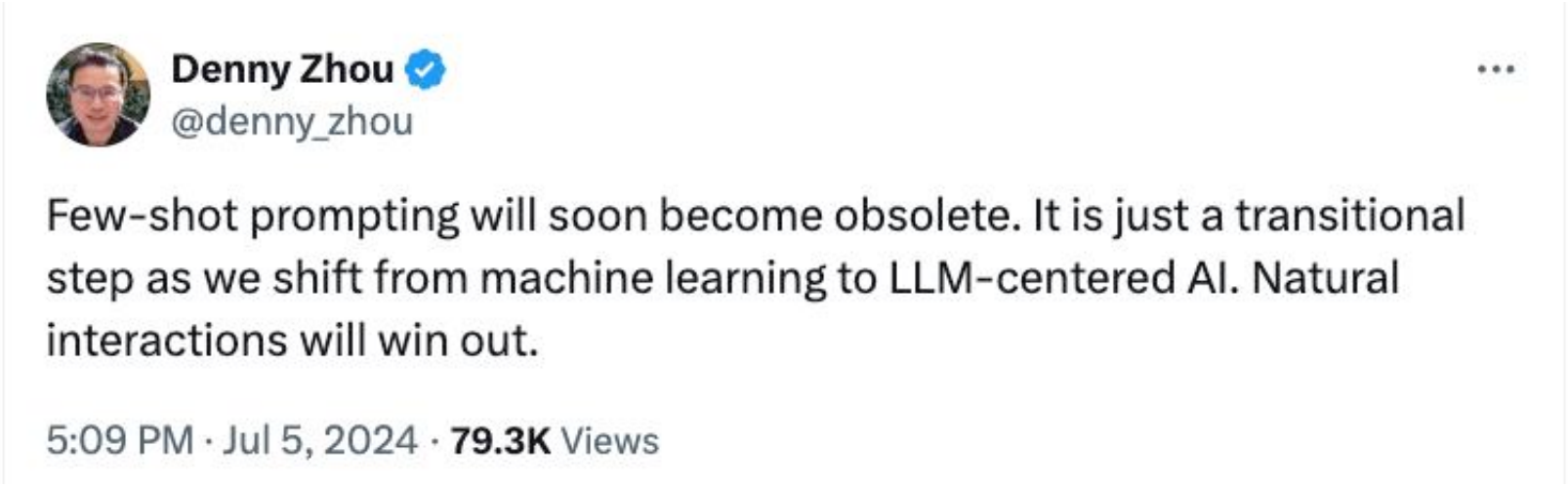
  "it is sometimes difficult to communicate the content or format of a task if no examples are given"
- **Zero-shot:** avoidance of spurious correlation, "unfairly hard"

  "at least some settings zero-shot is closest to how humans perform tasks"

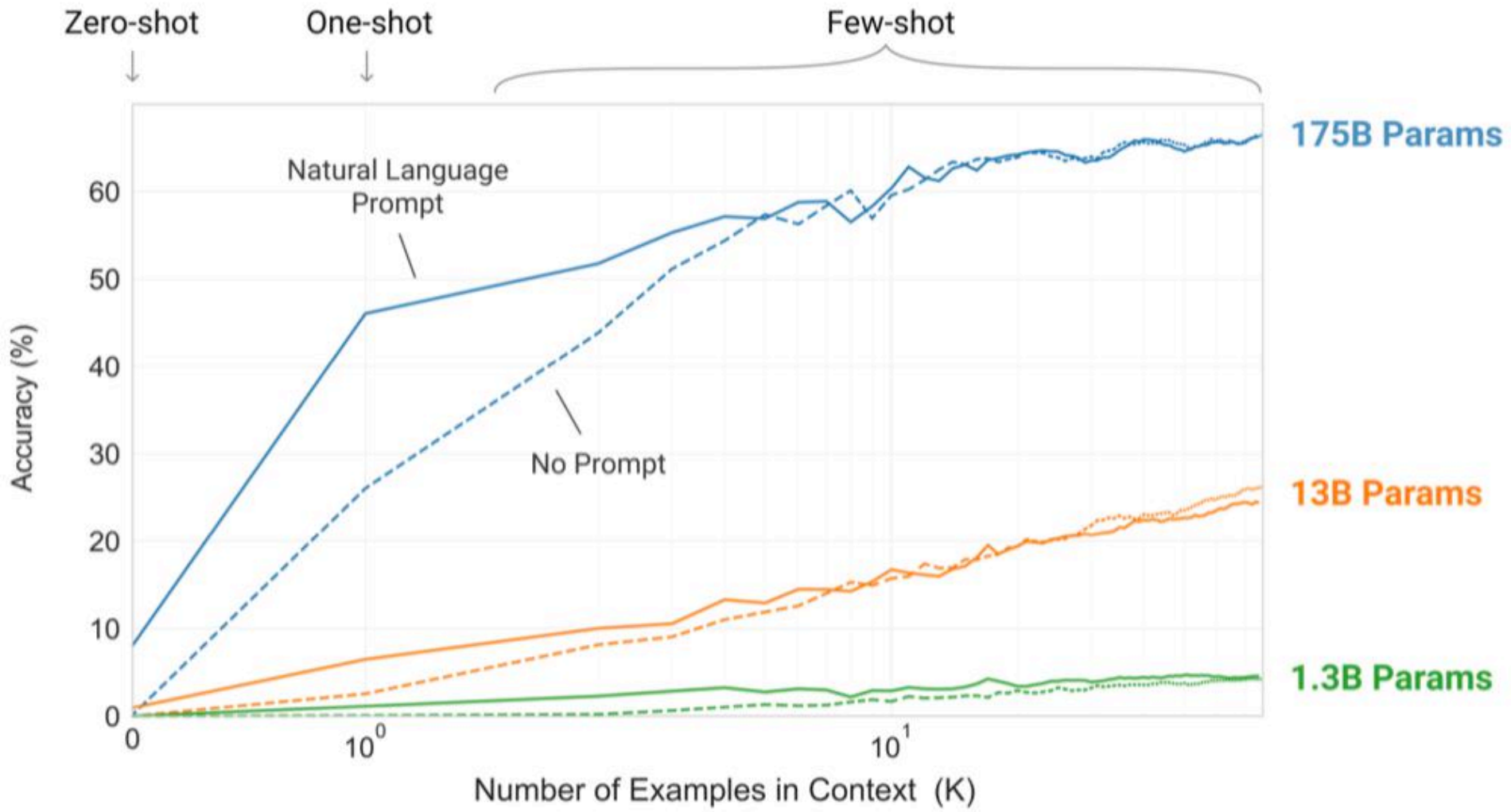
Approach

- **Few-shot:** stronger performance, only slightly behind state-of-the-art fine-tuned models

> "however, one-shot, or even sometimes zero-shot, seem like the fairest comparisons to human performance, and **are important targets for future work.**"



A summary of results



Evaluation tasks

- Tasks similar to language modeling
- Closed-book question answering
- Machine translation
- Winograd schema and commonsense reasoning
- Reading comprehension
- SuperGLUE
- NLI
- Novel tasks: on-the-fly reasoning, adaptation, open-ended text synthesis

Evaluation protocol

- Open-ended generation: beam search (size = 4), length penalty (α = 0.6)
- Multiple choices questions (MCQ):
  - K In-context examples (context + correct completion) + query context
  - Feed each answer choice separately and compare per-token likelihood
  - Additional benefits: $\frac{P(\text{completion}|\text{context})}{P(\text{completion}|\text{answer\_context})}$

- Yes/no questions: use True/False instead of 0/1

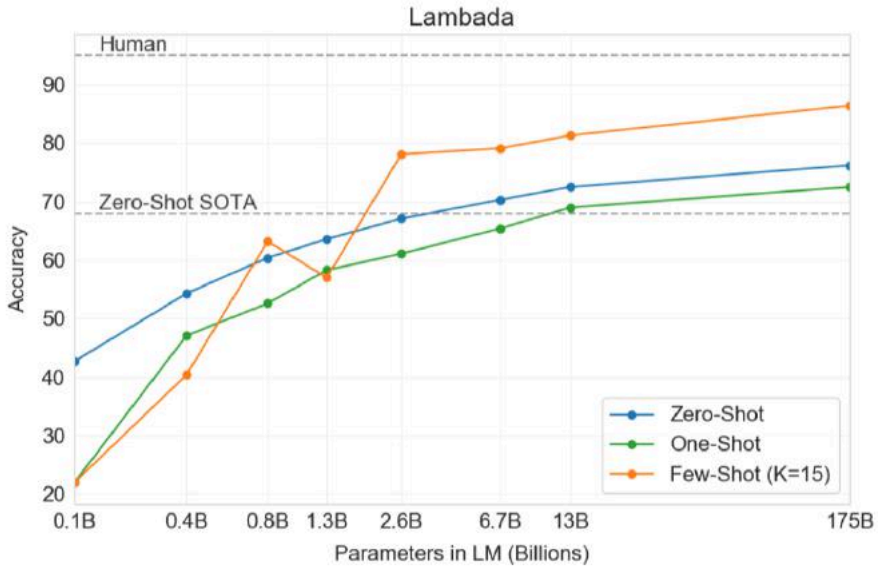
Published as a conference paper at ICLR 2024

LARGE LANGUAGE MODELS ARE NOT ROBUST MULTIPLE CHOICE SELECTORS

Chujie Zheng[†] **Hao Zhou**[‡] **Fandong Meng**[‡] **Jie Zhou**[‡] **Minlie Huang**[†*]
[†]The CoAI Group, DCST, BNRist, Tsinghua University, Beijing 100084, China
[‡]Pattern Recognition Center, WeChat AI, Tencent Inc., China
chujiezhengchn@gmail.com aihuang@tsinghua.edu.cn

Language modeling

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



LAMBADA

Context: He shook his head, took a step back and held his hands up as he tried to smile without losing a cigarette. "Yes you can," Julia said in a reassuring voice. "I 've already focused on my friend. You just have to click the shutter, on top, here."

Target sentence: He nodded sheepishly, through his cigarette away and took the _____.

Target word: camera

(Paperno et al., 2016)

Language modeling

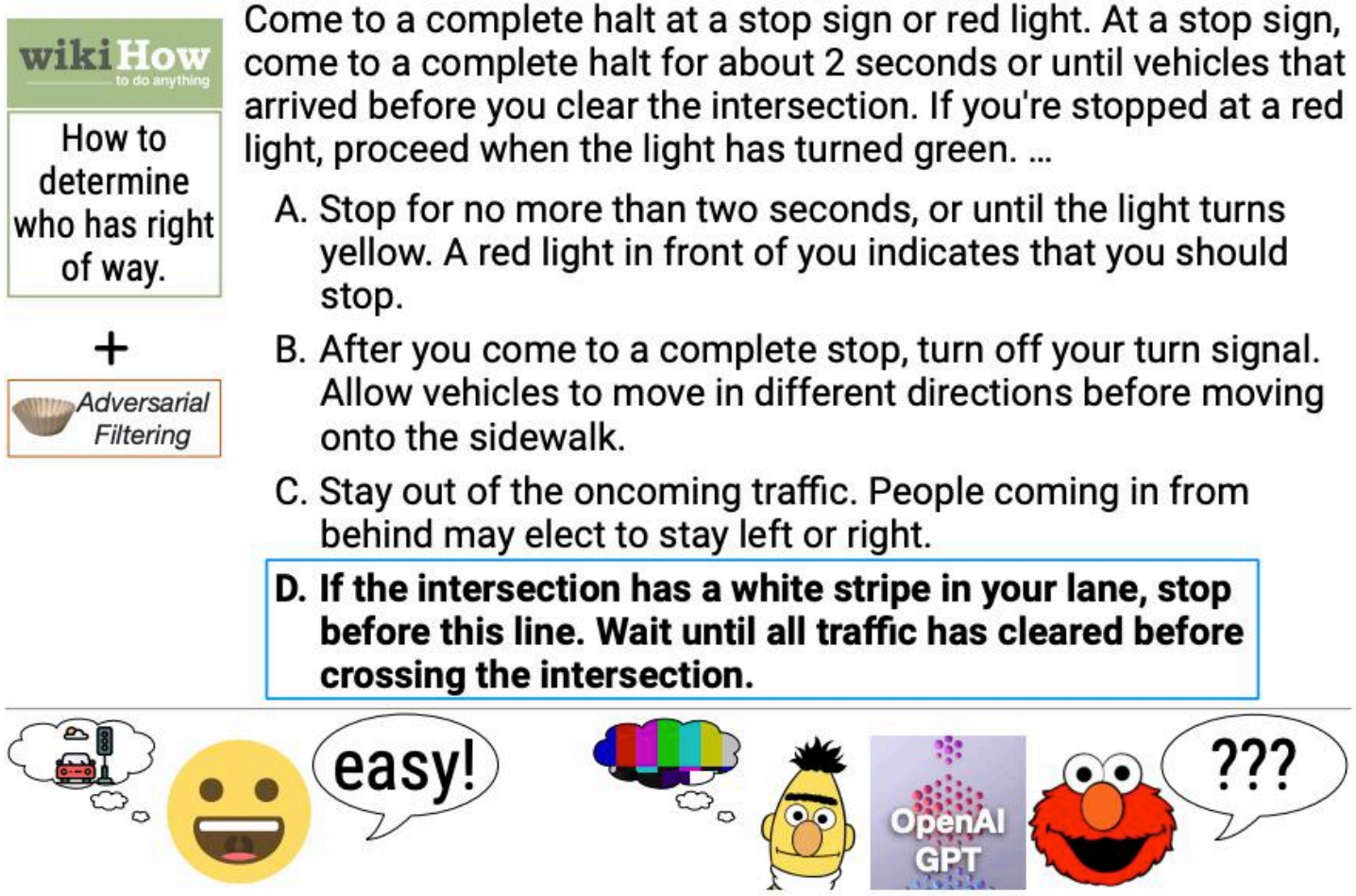
STORYCLOZE

Context	Right Ending	Wrong Ending
Karen was assigned a roommate her first year of college. Her roommate asked her to go to a nearby city for a concert. Karen agreed happily. The show was absolutely exhilarating.	Karen became good friends with her roommate.	Karen hated her roommate.
Jim got his first credit card in college. He didn't have a job so he bought everything on his card. After he graduated he amounted a $10,000 debt. Jim realized that he was foolish to spend so much money.	Jim decided to devise a plan for repayment.	Jim decided to open another credit card.
Gina misplaced her phone at her grandparents. It wasn't anywhere in the living room. She realized she was in the car before. She grabbed her dad's keys and ran outside.	She found her phone in the car.	She didn't want her phone anymore.

(Mostafazadeh et al., 2016)

Language modeling

HELLASWAG



(Zellers et al., 2019)

Open-domain question answering

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**

- Open-book vs closed-book QA

Question	Generated Answer	Correct	Probability
Who wrote the book the origin of species?	Charles Darwin	✓	83.4%
Who is the founder of the ubuntu project?	Mark Shuttleworth	✓	82.0%
Who is the quarterback for the green bay packers?	Aaron Rodgers	✓	81.1%
Panda is a national animal of which country?	China	✓	76.8%
Who came up with the theory of relativity?	Albert Einstein	✓	76.4%
When was the first star wars film released?	1977	✓	71.4%
What is the most common blood type in sweden?	A	✗	70.6%
Who is regarded as the founder of psychoanalysis?	Sigmund Freud	✓	69.3%
Who took the first steps on the moon in 1969?	Neil Armstrong	✓	66.8%
Who is the largest supermarket chain in the uk?	Tesco	✓	65.3%
What is the meaning of shalom in english?	peace	✓	64.0%
Who was the author of the art of war?	Sun Tzu	✓	59.6%
Largest state in the us by land mass?	California	✗	59.2%
Green algae is an example of which type of reproduction?	parthenogenesis	✗	56.5%

Machine translation

- GPT-3's training data: 93% English (by word count)

unsupervised NMT

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

Winograd-style and commonsense reasoning

Setting	Winograd	Winogrande (XL)
Fine-tuned SOTA	**90.1**[a]	**84.6**[b]
GPT-3 Zero-Shot	88.3*	70.2
GPT-3 One-Shot	89.7*	73.2
GPT-3 Few-Shot	88.6*	77.7

- **Example:** Grace was happy to trade me her sweater for my jacket. She thinks the [sweater | jacket] looks dowdy to her

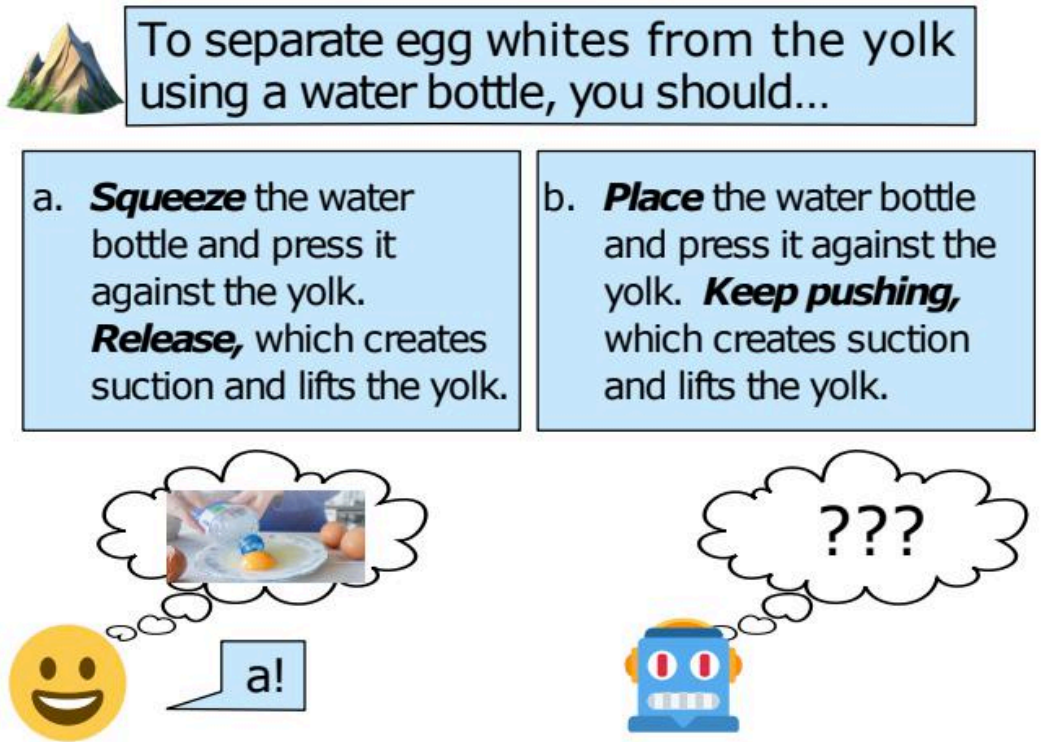
Correct Context →	Grace was happy to trade me her sweater for my jacket. She thinks the sweater
Incorrect Context →	Grace was happy to trade me her sweater for my jacket. She thinks the jacket
Target Completion →	looks dowdy on her.

Figure G.13: Formatted dataset example for Winograd. The 'partial' evaluation method we use compares the probability of the completion given a correct and incorrect context.

Winograd-style and commonsense reasoning

Setting	PIQA	ARC (Easy)	ARC (Challenge)	OpenBookQA
Fine-tuned SOTA	79.4	**92.0**[KKS+20]	**78.5**[KKS+20]	**87.2**[KKS+20]
GPT-3 Zero-Shot	**80.5***	68.8	51.4	57.6
GPT-3 One-Shot	**80.5***	71.2	53.2	58.8
GPT-3 Few-Shot	**82.8***	70.1	51.5	65.4

PIQA (PHYSICAL QA)



(Bisk et al., 2019)

Winograd-style and commonsense reasoning

Setting	PIQA	ARC (Easy)	ARC (Challenge)	OpenBookQA
Fine-tuned SOTA	79.4	**92.0**[KKS+20]	**78.5**[KKS+20]	**87.2**[KKS+20]
GPT-3 Zero-Shot	**80.5***	68.8	51.4	57.6
GPT-3 One-Shot	**80.5***	71.2	53.2	58.8
GPT-3 Few-Shot	**82.8***	70.1	51.5	65.4

- ARC: 3rd to 9th grade science exams

Knowledge Type	Example
Definition	What is a worldwide increase in temperature called? (A) greenhouse effect (B) global warming (C) ozone depletion (D) solar heating
Basic Facts & Properties	Which element makes up most of the air we breathe? (A) carbon (B) nitrogen (C) oxygen (D) argon
Structure	The crust, the mantle, and the core are structures of Earth. Which description is a feature of Earth's mantle? (A) contains fossil remains (B) consists of tectonic plates (C) is located at the center of Earth (D) has properties of both liquids and solids
Processes & Causal	What is the first step of the process in the formation of sedimentary rocks? (A) erosion (B) deposition (C) compaction (D) cementation
Teleology / Purpose	What is the main function of the circulatory system? (1) secrete enzymes (2) digest proteins (3) produce hormones (4) transport materials
Algebraic	If a red flowered plant (RR) is crossed with a white flowered plant (rr), what color will the offspring be? (A) 100% pink (B) 100% red (C) 50% white, 50% red (D) 100% white
Experiments	Scientists perform experiments to test hypotheses. How do scientists try to remain objective during experiments? (A) Scientists analyze all results. (B) Scientists use safety precautions. (C) Scientists conduct experiments once. (D) Scientists change at least two variables.
Spatial / Kinematic	In studying layers of rock sediment, a geologist found an area where older rock was layered on top of younger rock. Which best explains how this occurred? (A) Earthquake activity folded the rock layers...

Reading comprehension

Setting	CoQA	DROP	QuAC	SQuADv2	RACE-h	RACE-m
Fine-tuned SOTA	**90.7**[a]	**89.1**[b]	**74.4**[c]	**93.0**[d]	**90.0**[e]	**93.1**[e]
GPT-3 Zero-Shot	81.5	23.6	41.5	59.5	45.5	58.4
GPT-3 One-Shot	84.0	34.3	43.3	65.4	45.9	57.4
GPT-3 Few-Shot	85.0	36.5	44.3	69.8	46.8	58.1

Subtraction (28.8%)	That year, his **Untitled (1981)**, a painting of a haloed, black-headed man with a bright red skeletal body, depicted amid the artists signature scrawls, was **sold by Robert Lehrman for $16.3 million, well above its $12 million high estimate**.	How many more dollars was the Untitled (1981) painting sold for than the 12 million dollar estimation?	4300000
Comparison (18.2%)	In **1517, the seventeen-year-old King sailed to Castile**. There, his Flemish court .... **In May 1518, Charles traveled to Barcelona in Aragon**.	Where did Charles travel to first, Castile or Barcelona?	Castile
Selection (19.4%)	In 1970, to commemorate the 100th anniversary of the founding of Baldwin City, **Baker University professor and playwright Don Mueller and Phyllis E. Braun, Business Manager, produced a musical play entitled The Ballad Of Black Jack** to tell the story of the events that led up to the battle.	Who was the University professor that helped produce The Ballad Of Black Jack, Ivan Boyd or Don Mueller?	Don Mueller

DROP (Dua et al., 2019)

What did the General Conference on Weights and Measures name after Tesla in 1960?

Ground Truth Answers: SI unit of magnetic flux density

Tesla was renowned for his achievements and showmanship, eventually earning him a reputation in popular culture as an archetypal "mad scientist". His patents earned him a considerable amount of money, much of which was used to finance his own projects with varying degrees of success.:121,154 He lived most of his life in a series of New York hotels, through his retirement. Tesla died on 7 January 1943. His work fell into relative obscurity after his death, but in 1960 the General Conference on Weights and Measures named the SI unit of magnetic flux density the tesla in his honor. There has been a resurgence in popular interest in Tesla since the 1990s.

SQuAD (Rajpurkar et al., 2017)

Reading comprehension

Setting	CoQA	DROP	QuAC	SQuADv2	RACE-h	RACE-m
Fine-tuned SOTA	**90.7**[a]	**89.1**[b]	**74.4**[c]	**93.0**[d]	**90.0**[e]	**93.1**[e]
GPT-3 Zero-Shot	81.5	23.6	41.5	59.5	45.5	58.4
GPT-3 One-Shot	84.0	34.3	43.3	65.4	45.9	57.4
GPT-3 Few-Shot	85.0	36.5	44.3	69.8	46.8	58.1

Passage:
In a small village in England about 150 years ago, a mail coach was standing on the street. It didn't come to that village often. People had to pay a lot to get a letter. The person who sent the letter didn't have to pay the postage, while the receiver had to.
"Here's a letter for Miss Alice Brown," said the mailman.
" I'm Alice Brown," a girl of about 18 said in a low voice.
Alice looked at the envelope for a minute, and then handed it back to the mailman.
"I'm sorry I can't take it, I don't have enough money to pay it", she said.
A gentleman standing around were very sorry for her. Then he came up and paid the postage for her.
When the gentleman gave the letter to her, she said with a smile, " Thank you very much, This letter is from Tom. I'm going to marry him. He went to London to look for work. I've waited a long time for this letter, but now I don't need it, there is nothing in it."
"Really? How do you know that?" the gentleman said in surprise.
"He told me that he would put some signs on the envelope. Look, sir, this cross in the corner means that he is well and this circle means he has found work. That's good news."
The gentleman was Sir Rowland Hill. He didn't forgot Alice and her letter.
"The postage to be paid by the receiver has to be changed," he said to himself and had a good plan.
"The postage has to be much lower, what about a penny? And the person who sends the letter pays the postage. He has to buy a stamp and put it on the envelope." he said . The government accepted his plan. Then the first stamp was put out in 1840. It was called the "Penny Black". It had a picture of the Queen on it.

Questions:

1): The first postage stamp was made _.
A. in England B. in America C. by Alice D. in 1910

2): The girl handed the letter back to the mailman because _.
A. she didn't know whose letter it was
B. she had no money to pay the postage
C. she received the letter but she didn't want to open it
D. she had already known what was written in the letter

3): We can know from Alice's words that _.
A. Tom had told her what the signs meant before leaving
B. Alice was clever and could guess the meaning of the signs
C. Alice had put the signs on the envelope herself
D. Tom had put the signs as Alice had told him to

4): The idea of using stamps was thought of by _.
A. the government
B. Sir Rowland Hill
C. Alice Brown
D. Tom

5): From the passage we know the high postage made _.
A. people never send each other letters
B. lovers almost lose every touch with each other
C. people try their best to avoid paying it
D. receivers refuse to pay the coming letters

Answer: ADABC

RACE (Lai et al., 2017)

- Reading comprehension tests for middle and high school Chinese students (age between 12 and 18)

Reading comprehension

Context →	Helsinki is the capital and largest city of Finland. It is in the region of Uusimaa, in southern Finland, on the shore of the Gulf of Finland. Helsinki has a population of , an urban population of , and a metropolitan population of over 1.4 million, making it the most populous municipality and urban area in Finland. Helsinki is some north of Tallinn, Estonia, east of Stockholm, Sweden, and west of Saint Petersburg, Russia. Helsinki has close historical connections with these three cities. The Helsinki metropolitan area includes the urban core of Helsinki, Espoo, Vantaa, Kauniainen, and surrounding commuter towns. It is the world's northernmost metro area of over one million people, and the city is the northernmost capital of an EU member state. The Helsinki metropolitan area is the third largest metropolitan area in the Nordic countries after Stockholm and Copenhagen, and the City of Helsinki is the third largest after Stockholm and Oslo. Helsinki is Finland's major political, educational, financial, cultural, and research center as well as one of northern Europe's major cities. Approximately 75% of foreign companies that operate in Finland have settled in the Helsinki region. The nearby municipality of Vantaa is the location of Helsinki Airport, with frequent service to various destinations in Europe and Asia. Q: what is the most populous municipality in Finland? A: Helsinki Q: how many people live there? A: 1.4 million in the metropolitan area Q: what percent of the foreign companies that operate in Finland are in Helsinki? A: 75% Q: what towns are a part of the metropolitan area? A:
Target Completion →	Helsinki, Espoo, Vantaa, Kauniainen, and surrounding commuter towns

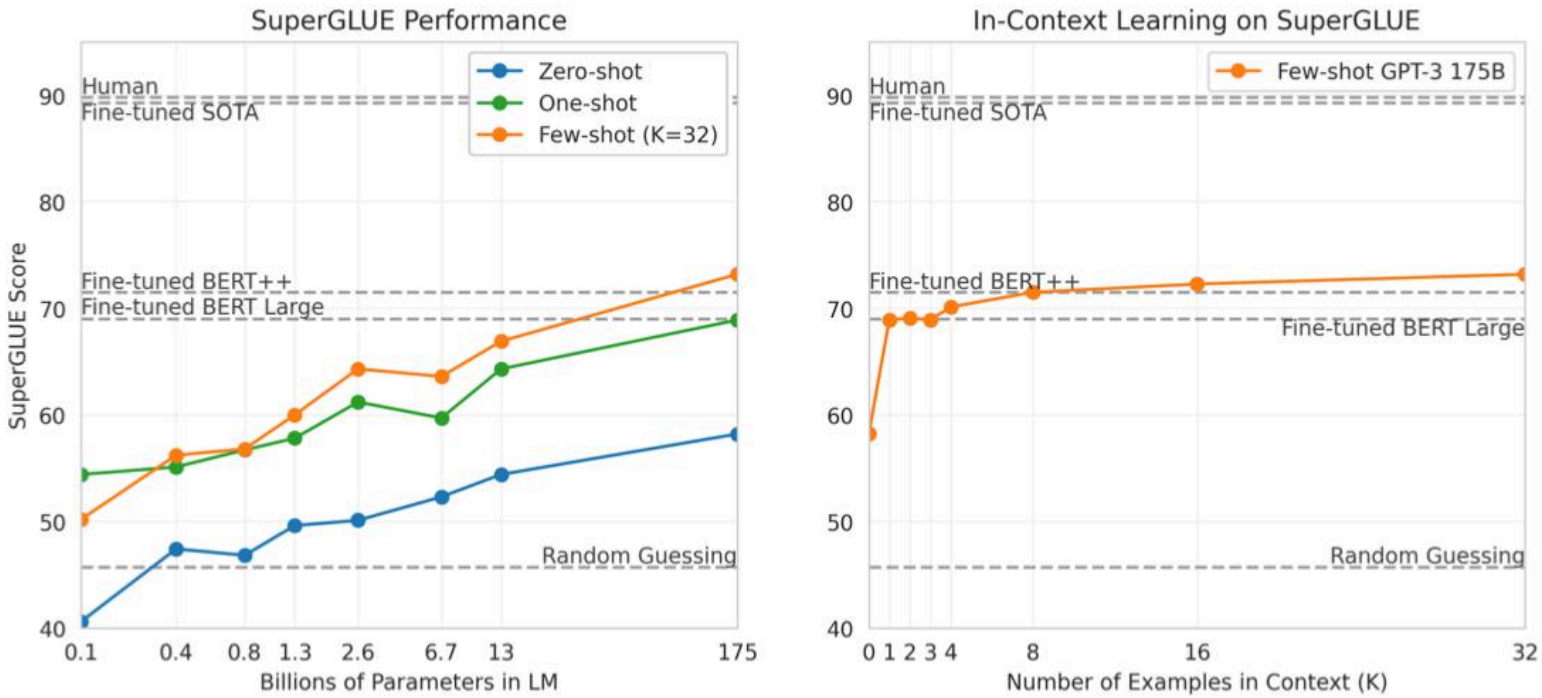
Figure G.18: Formatted dataset example for CoQA

CoQA (Reddy et al., 2019)

SuperGLUE

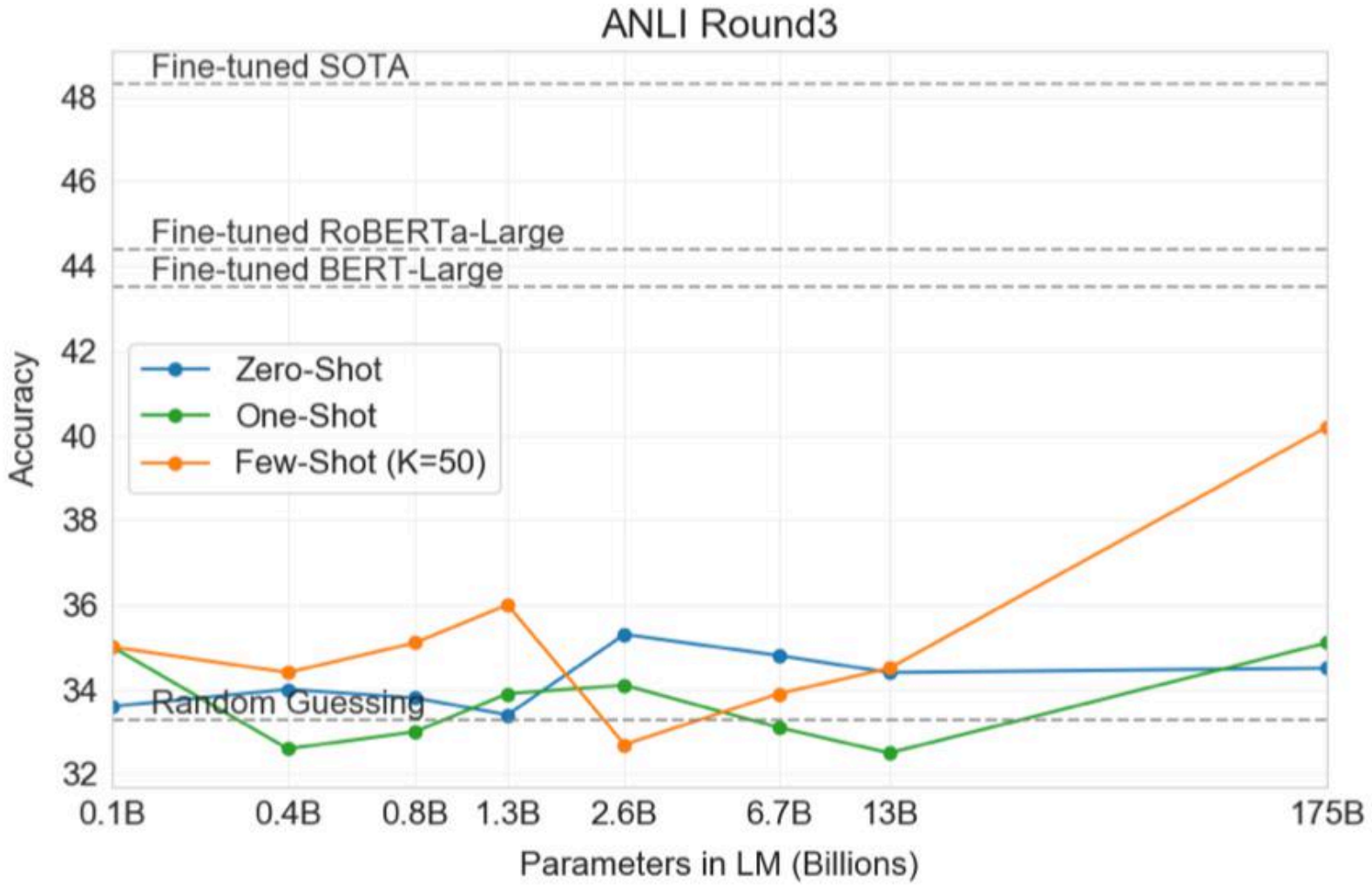
	SuperGLUE Average	BoolQ Accuracy	CB Accuracy	CB F1	COPA Accuracy	RTE Accuracy
Fine-tuned SOTA	**89.0**	**91.0**	**96.9**	**93.9**	**94.8**	**92.5**
Fine-tuned BERT-Large	69.0	77.4	83.6	75.7	70.6	71.7
GPT-3 Few-Shot	71.8	76.4	75.6	52.0	92.0	69.0

	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



SuperGLUE (Wang et al., 2019)

Natural language inference (NLI)



ANLI (Nie et al., 2019)

Novel tasks

- Arithmetic
- Word scrambling and manipulation
- SAT analogies
- News article generation
- Learning and using novel words

Why synthetic tasks?

- Easier to control, scale and manipulate
- Less data contamination
- Sometimes provides very clear insights of what is going on

Novel tasks

Context →	Please unscramble the letters into a word, and write that word: asinoc =
Target Completion →	casino

Figure G.19: Formatted dataset example for Cycled Letters

Context →	Please unscramble the letters into a word, and write that word: r e!c.i p r o.c a/l =
Target Completion →	reciprocal

Figure G.26: Formatted dataset example for Symbol Insertion

Context →	Please unscramble the letters into a word, and write that word: taefed =
Target Completion →	defeat

Figure G.27: Formatted dataset example for Reversed Words

Novel tasks

Context →	Q: What is 98 plus 45? A:
Target Completion →	143

Figure G.44: Formatted dataset example for Arithmetic 2D+

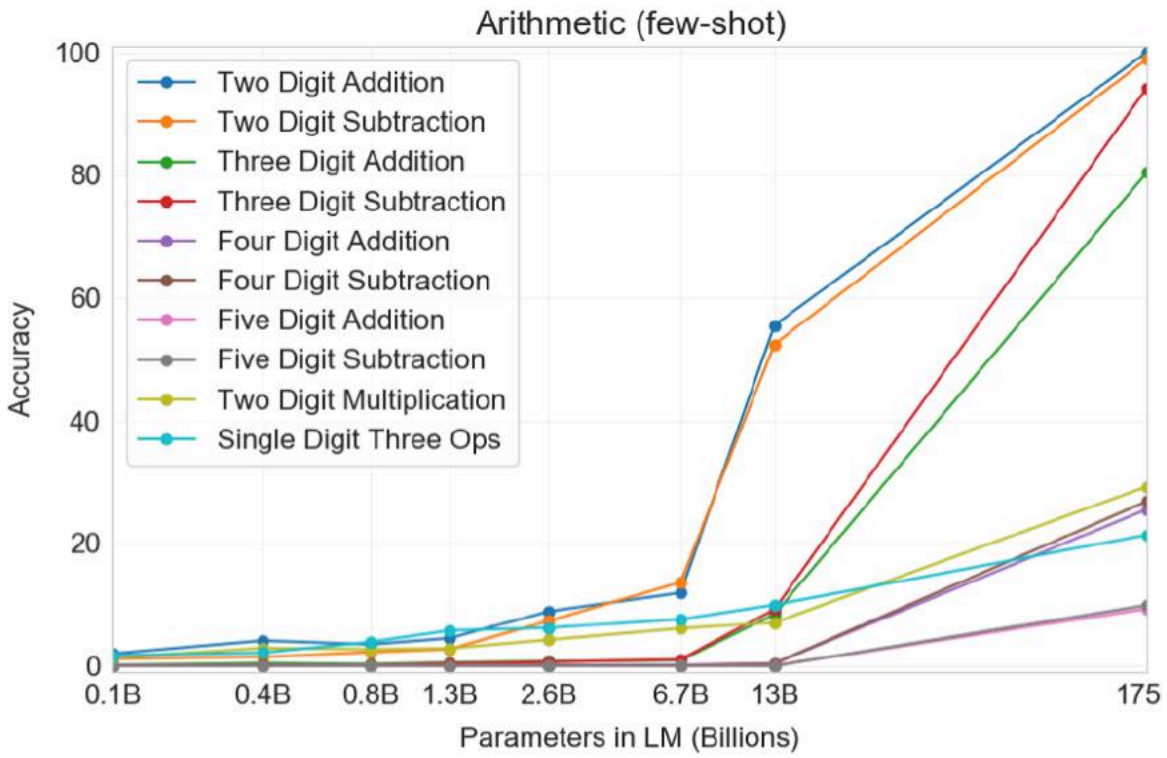
Context →	Q: What is 6209 minus 3365? A:
Target Completion →	2844

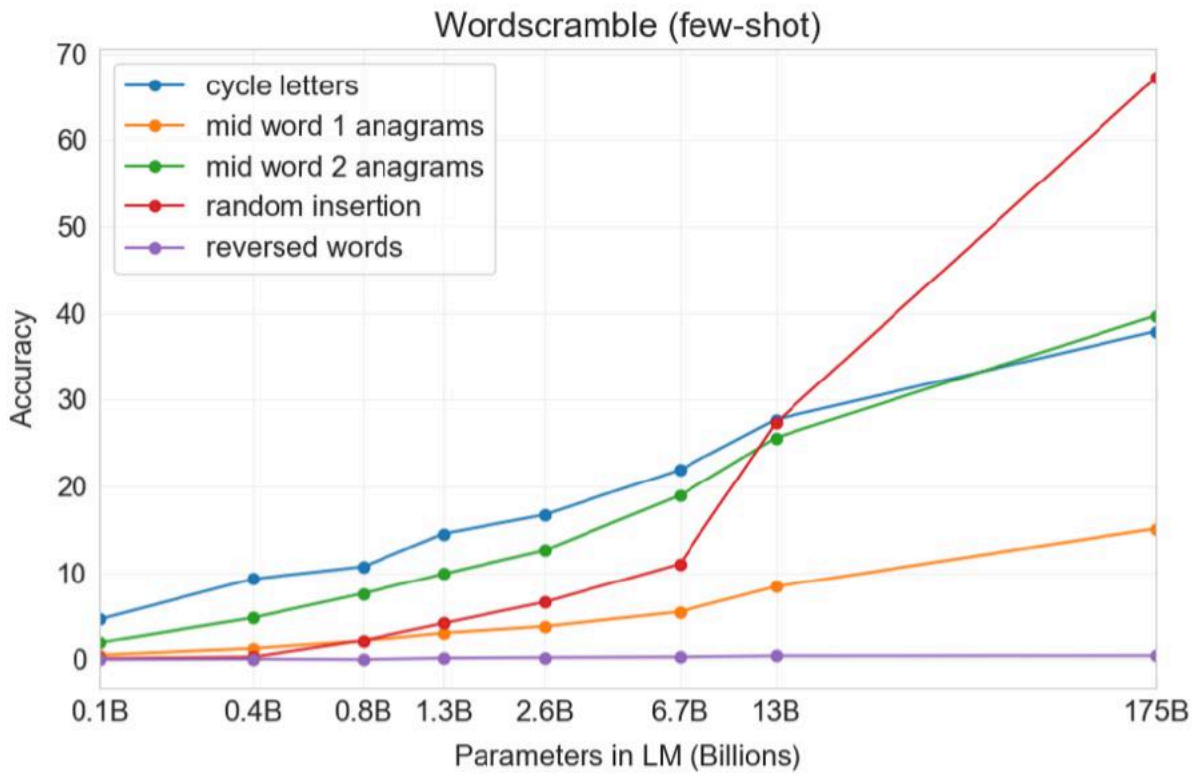
Figure G.48: Formatted dataset example for Arithmetic 4D-

Context →	lull is to trust as
Correct Answer →	cajole is to compliance
Incorrect Answer →	balk is to fortitude
Incorrect Answer →	betray is to loyalty
Incorrect Answer →	hinder is to destination
Incorrect Answer →	soothe is to passion

Figure G.12: Formatted dataset example for SAT Analogies

Novel tasks

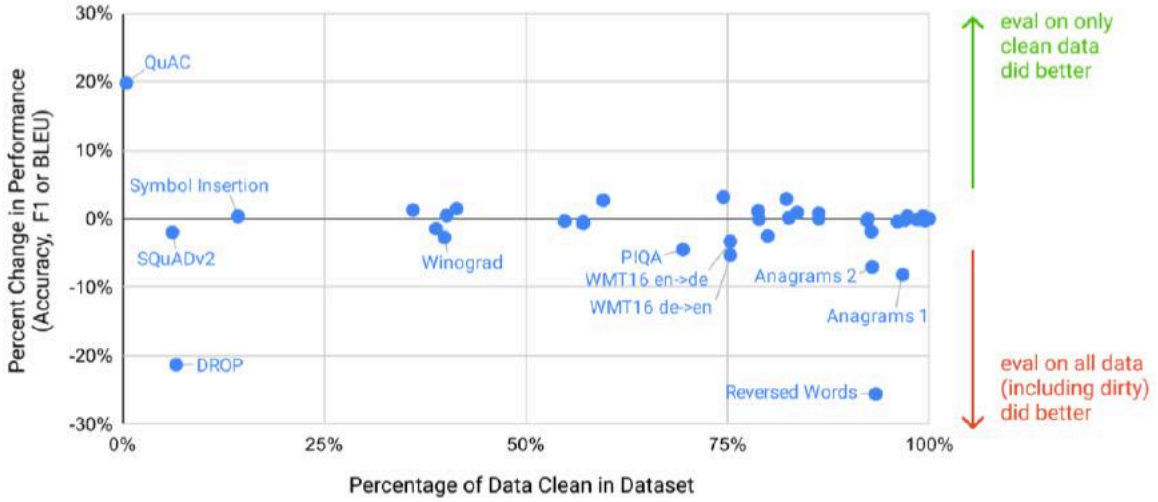

Arithmetic (few-shot)
Two Digit Addition
Two Digit Subtraction
Three Digit Addition
Three Digit Subtraction
Four Digit Addition
Four Digit Subtraction
Five Digit Addition
Five Digit Subtraction
Two Digit Multiplication
Single Digit Three Ops
Accuracy
100
80
60
40
20
0
0.1B
0.4B
0.8B
1.3B
2.6B
6.7B
13B
175
Parameters in LM (Billions)


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)

Contamination analysis

- How to decide which examples are contaminated?
  - "defined roughly as examples that have a 13-gram overlap with anything in the pretraining set"
- How to decide estimated performance gains from contamination?
  - Compare the performance on the "clean" subset vs entire dataset

A major methodological concern with language models pretrained on a broad swath of internet data, particularly large models with the capacity to memorize vast amounts of content, is potential contamination of downstream tasks by having their test or development sets inadvertently seen during pre-training. To reduce such contamination, we searched for and attempted to remove any overlaps with the development and test sets of all benchmarks studied in this paper. Unfortunately, a bug in the filtering caused us to ignore some overlaps, and due to the cost of training it was not feasible to retrain the model. In Section 4 we characterize the impact of the remaining overlaps, and in future work we will more aggressively remove data contamination.



Understanding In-Context Learning

Understanding in-context learning

Extrapolating to Unnatural Language Processing with GPT-3's In-context Learning: The Good, the Bad, and the Mysterious

Frieda Rong

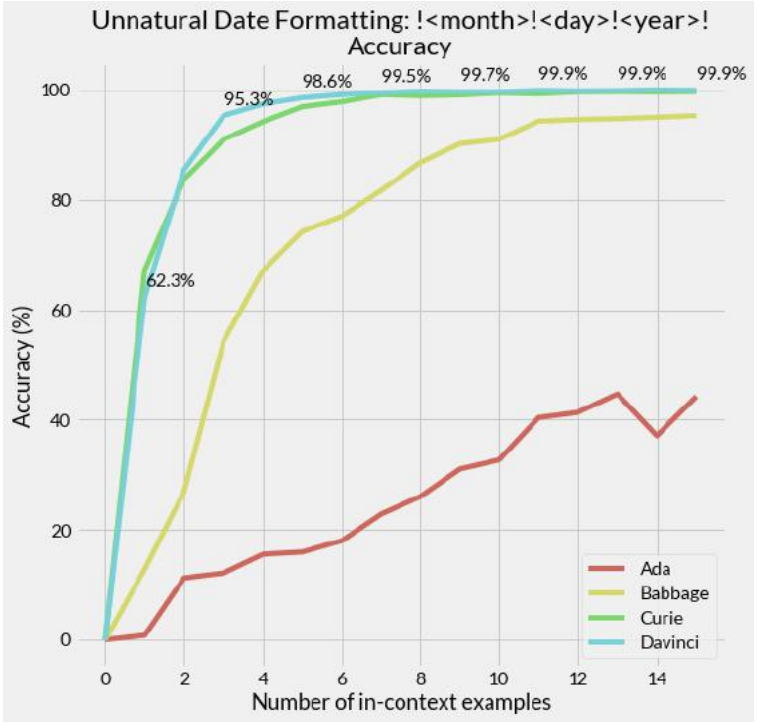
May 28, 2021

```
Input: 2014-06-01
Output: !06!01!2014!
Input: 2007-12-13
Output: !12!13!2007!
Input: 2010-09-23
Output: !09!23!2010!
Input: 2005-07-23
Output: !07!23!2005!
```

in-context examples

test example

model completion



http://ai.stanford.edu/blog/in-context-learning/

Understanding in-context learning

- **Hypothesis #1:** Transformers perform implicit gradient descent to update an "inner model"

Transformers Learn In-Context by Gradient Descent

Johannes von Oswald [1 2] Eyvind Niklasson [2] Ettore Randazzo [2] João Sacramento [1]
Alexander Mordvintsev [2] Andrey Zhmoginov [2] Max Vladymyrov [2]

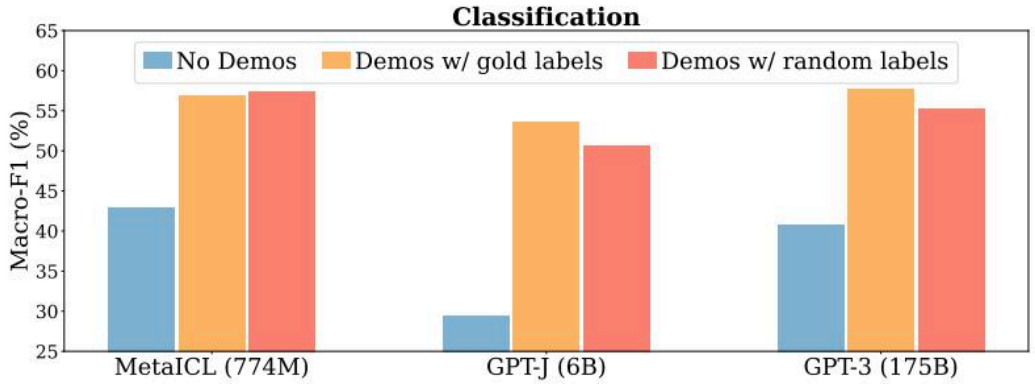
Why Can GPT Learn In-Context?
Language Models Implicitly Perform Gradient Descent as Meta-Optimizers

Damai Dai[†,*] Yutao Sun[‖,*] Li Dong[‡] Yaru Hao[‡] Shuming Ma[‡] Zhifang Sui[†] Furu Wei[‡]

- **Hypothesis #2:** Transformers learn tasks required for downstream applications during pre-training, and in-context demonstrations are only used to recognize which task is required

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
{sewon,alrope,ahai,hannaneh,lsz}@cs.washington.ed
{artetxe,mikelewis}@meta.com



Ground-truth labels don't matter!

Understanding in-context learning

Disentangle In-context learning into two roles – **task recognition (TR)** vs **task learning (TL)**

- TR: recognizes the task from demonstrations and applies LLMs' pre-trained priors
- TL: learns a new input-label mapping from demonstrations
- ICL performs both TR and TL, but TL emerges with **larger models** and **more demonstrations**

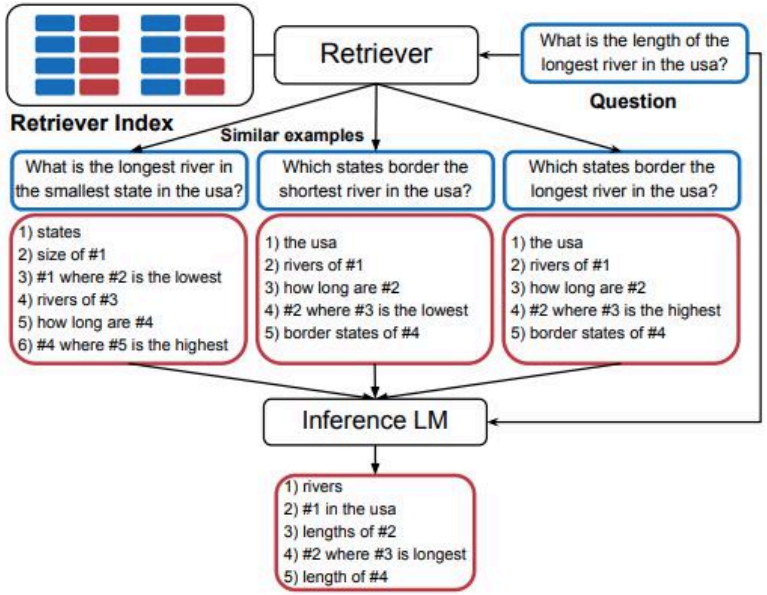
Pan et al. What In-Context Learning "Learns" In-Context: Disentangling Task Recognition and Task Learning, 2023

Improving in-context learning performance

- Instead of randomly sampling K in-context examples, you should use "high-quality" and similar ones!

Learning To Retrieve Prompts for In-Context Learning

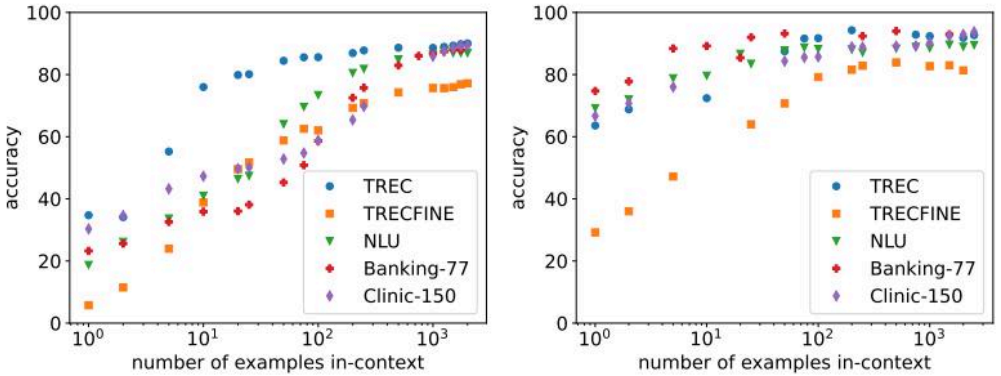
Ohad Rubin **Jonathan Herzig** **Jonathan Berant**
The Blavatnik School of Computer Science, Tel Aviv University
{ohad.rubin,jonathan.herzig,joberant}@cs.tau.ac.il



- Pack more examples in long-context models!

In-Context Learning with Long-Context Models: An In-Depth Exploration

Amanda Bertsch[γ] abertsch@cs.cmu.edu
Maor Ivgi [τ] maor.ivgi@cs.tau.ac.il
Uri Alon[γ*] urialon@cs.cmu.edu
Jonathan Berant[τ] joberant@cs.tau.ac.il
Matthew R. Gormley[γ] mgormley@cs.cmu.edu
Graham Neubig[γ] gneubig@cs.cmu.edu



(a) Using randomly selected examples. (b) Using retrieved examples.

Scaling Laws

The Bitter Lesson

Rich Sutton

March 13, 2019

The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin. The ultimate reason for this is Moore's law, or rather its generalization of continued exponentially falling cost per unit of computation. Most AI research has been conducted as if the computation available to the agent were constant (in which case leveraging human knowledge would be one of the only ways to improve performance) but, over a slightly longer time than a typical research project, massively more computation inevitably becomes available. Seeking an improvement that makes a difference in the shorter term, researchers seek to leverage their human knowledge of the domain, but the only thing that matters in the long run is the leveraging of computation. These two need not run counter to each other, but in practice they tend to. Time spent on one is time not spent on the other. There are psychological commitments to investment in one approach or the other. And the human-knowledge approach tends to complicate methods in ways that make them less suited to taking advantage of general methods leveraging computation. There were many examples of AI researchers' belated learning of this bitter lesson, and it is instructive to review some of the most prominent.

One thing that should be learned from the bitter lesson is the great power of general purpose methods, of methods that continue to scale with increased computation even as the available computation becomes very great. The two methods that seem to scale arbitrarily in this way are *search* and *learning*.

Simple question

Suppose you take a deep network, and you multiply its size by C_1 and its dataset size by C_2

By how much does the compute requirement (in FLOPs) increase?

FLOP= Floating point operation (addition/multiplication/ division)

Scaling up Deep Learning

CNNs drove initial successes, esp for vision datasets (CIFAR, ImageNet etc.)

What's a recipe to scale them up for arbitrary image tasks?

[Tan & Le '19] "Efficient (Conv)-Nets" : If you have 2^N factor more compute, scale up width, depth, image-size by α^N, β^N, γ^N where α, β, γ are determined by grid search on smaller conv-nets for the same task

Compute requirement for forward pass (Transformer)

(Factor 2 for multiply accumulate)

- Embeddings
  - $2 \times \text{seq\_len} \times \text{vocab\_size} \times \text{d\_model}$
- Attention (Single Layer)
  - **Key, query and value projections:** $2 \times 3 \times \text{seq\_len} \times \text{d\_model} \times (\text{key\_size} \times \text{num\_heads})$
  - **Key @ Query logits:** $2 \times \text{seq\_len} \times \text{seq\_len} \times (\text{key\_size} \times \text{num\_heads})$
  - **Softmax:** $3 \times \text{num\_heads} \times \text{seq\_len} \times \text{seq\_len}$
  - **Softmax @ query reductions:** $2 \times \text{seq\_len} \times \text{seq\_len} \times (\text{key\_size} \times \text{num\_heads})$
  - **Final Linear:** $2 \times \text{seq\_len} \times (\text{key\_size} \times \text{num\_heads}) \times \text{d\_model}$
- Dense Block (Single Layer)
  - $2 \times \text{seq\_len} \times (\text{d\_model} \times \text{ffw\_size} + \text{d\_model} \times \text{ffw\_size})$
- Final Logits
  - $2 \times \text{seq\_len} \times \text{d\_model} \times \text{vocab\_size}$

Total forward pass FLOPs: embeddings¸num_layers ¹total_attention¸dense_block° + logits

(In [Kaplan et al'20] approximated as 6ND; N =# parameters, D = # tokens)

Scaling Laws for Neural Language Models

Jared Kaplan *
Johns Hopkins University, OpenAI

Sam McCandlish*
OpenAI

et al, 2021

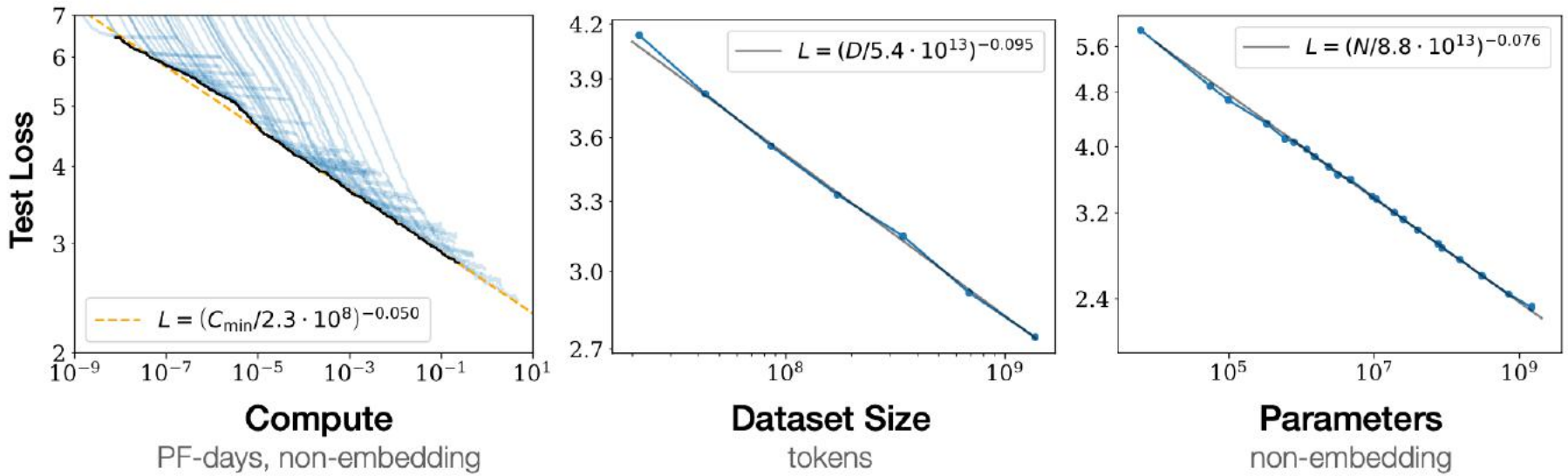


Figure 1 Language modeling performance improves smoothly as we increase the model size, datasetset size, and amount of compute[2] used for training. For optimal performance all three factors must be scaled up in tandem. Empirical performance has a power-law relationship with each individual factor when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

Claims that aged relatively well…

Performance depends strongly on scale, weakly on model shape: Model performance depends most strongly on scale, which consists of three factors: the number of model parameters N (excluding embeddings), the size of the dataset D, and the amount of compute C used for training. Within reasonable limits, performance depends very weakly on other architectural hyperparameters such as depth vs. width. (Section 3)

Smooth power laws: Performance has a power-law relationship with each of the three scale factors N, D, C when not bottlenecked by the other two, with trends spanning more than six orders of magnitude (see Figure 1). We observe no signs of deviation from these trends on the upper end, though performance must flatten out eventually before reaching zero loss. (Section 3)

Universality of overfitting: Performance improves predictably as long as we scale up N and D in tandem, but enters a regime of diminishing returns if either N or D is held fixed while the other increases. The

Universality of training: Training curves follow predictable power-laws whose parameters are roughly independent of the model size. By extrapolating the early part of a training curve, we can roughly predict the loss that would be achieved if we trained for much longer. (Section 5)

Transfer improves with test performance: When we evaluate models on text with a different distribution than they were trained on, the results are strongly correlated to those on the training validation set with a roughly constant offset in the loss – in other words, transfer to a different distribution incurs a constant penalty but otherwise improves roughly in line with performance on the training set. (Section 3.2.2)

Claims that did not …

Sample efficiency: Large models are more sample-efficient than small models, reaching the same level of performance with fewer optimization steps (Figure 2) and using fewer data points (Figure 4).

Convergence is inefficient: When working within a fixed compute budget C but without any other restrictions on the model size N or available data D, we attain optimal performance by training *very large models* and stopping *significantly short of convergence* (see Figure 3). Maximally compute-efficient training would therefore be far more sample efficient than one might expect based on training small models to convergence, with data requirements growing very slowly as $D \sim C^{0.27}$ with training compute. (Section 6)

Optimal batch size: 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 [MKAT18]; it is roughly 1-2 million tokens at convergence for the largest models we can train. (Section 5.1)

10× increase in compute should be allocated to a 5.5× increase in model size and a 1.8× increase in training tokens.

Brief Era of Undertrained Mega Models (2020-22)

Implication of Kaplan et al. [2020] : 10× increase in compute should be allocated to a 5.5× increase in model size and a 1.8× increase in training tokens.”

Gopher [Rae et al'21, Google]

280B parameters, 300B tokens…

Scaling Language Models: Methods, Analysis & Insights from Training *Gopher*

Model	Layers	Number Heads	Key/Value Size	d_{model}	Max LR	Batch Size
44M	8	16	32	512	6×10^{-4}	0.25M
117M	12	12	64	768	6×10^{-4}	0.25M
417M	12	12	128	1,536	2×10^{-4}	0.25M
1.4B	24	16	128	2,048	2×10^{-4}	0.25M
7.1B	32	32	128	4,096	1.2×10^{-4}	2M
Gopher 280B	80	128	128	16,384	4×10^{-5}	3M → 6M

Table 1 | **Model architecture details.** For each model, we list the number of layers, the key/value size, the bottleneck activation size d_{model}, the maximum learning rate, and the batch size. The feed-forward size is always $4 \times d_{model}$.

PaLM [Choudhery et al'22]

PaLM: Scaling Language Modeling with Pathways
[PaLM2] was a followup

540 B parameters; 780B tokens

Design tailored for parallelization in TPU v4 pod client-server architecture (Pathways)

Later stages use bigger batch sizes for better gradient estimate (less noise)

Model	# of Parameters (in billions)	Accelerator chips	Model FLOPS utilization
GPT-3	175B	V100	21.3%
Gopher	280B	4096 TPU v3	32.5%
Megatron-Turing NLG	530B	2240 A100	30.2%
PaLM	540B	6144 TPU v4	46.2%

Model	Layers	# of Heads	d_{model}	# of Parameters (in billions)	Batch Size
PaLM 8B	32	16	4096	8.63	$256 \rightarrow 512$
PaLM 62B	64	32	8192	62.50	$512 \rightarrow 1024$
PaLM 540B	118	48	18432	540.35	$512 \rightarrow 1024 \rightarrow 2048$

Table 1: Model architecture details. We list the number of layers, d_{model}, the number o attention heads and attention head size. The feed-forward size d_{ff} is always $4 \times d_{\text{model}}$ an attention head size is always 256.

Deterministic batches; "fully bitwise reproducible from any checkpoint".

PaLM (the hardware)

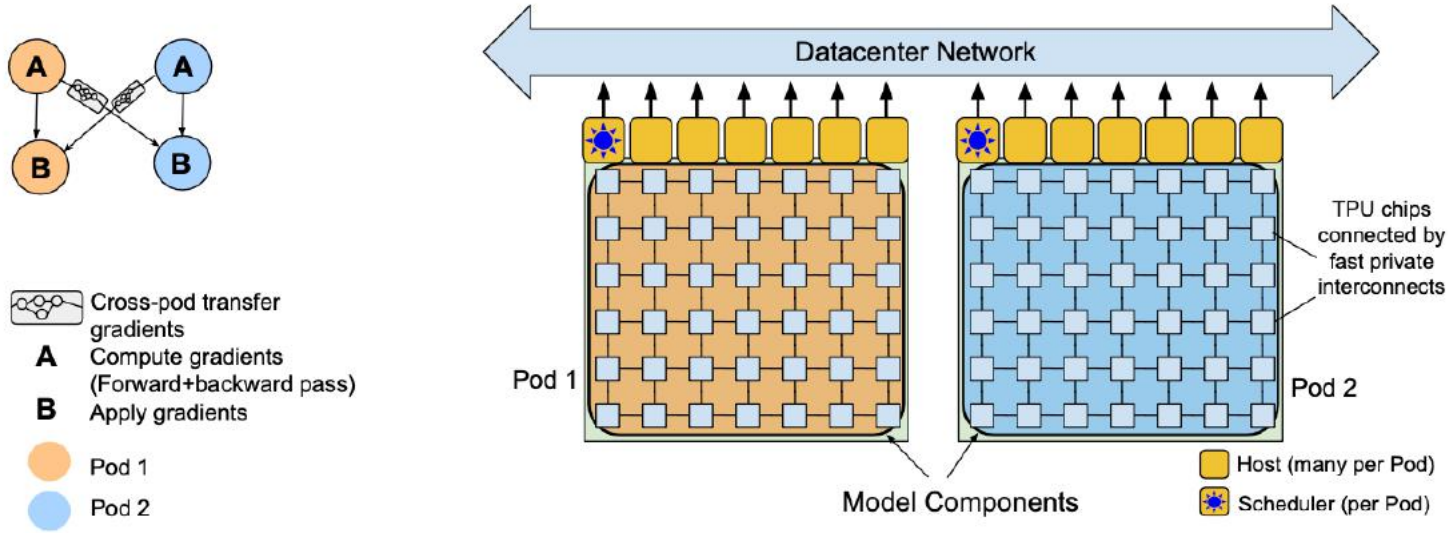


Figure 2: The Pathways system (Barham et al., 2022) scales training across two TPU v4 pods using two-way data parallelism at the pod level.

Figure 2 shows how the Pathways system executes the two-way pod-level data parallelism. A single Python client constructs a sharded dataflow program (shown on the left in Figure 2) that launches JAX/XLA (XLA, 2019) work on remote servers that each comprise a TPU pod. The program contains a component A for within-pod forward+backward computation (including within-pod gradient reduction), transfer subgraph for cross-pod gradient transfer, and a component B for optimizer update (including summation of local and remote gradients).

Megatron Tuning NLG (Nvidia, 2022)

530B parameters, 270B tokens

monolithic (unlike Google PaLM, PaLM2); served to highlight Nvidia's own parallelism solution (NVLink within a node, InfiniBand across nodes)*

In hindsight, a fairly unexceptional effort….

By combining tensor-slicing and pipeline parallelism, we can operate them within the regime where they are most effective. More specifically, the system uses tensor-slicing from Megatron-LM to scale the model within a node and uses pipeline parallelism from DeepSpeed to scale the model across nodes.

Chinchilla (DeepMind)

- DeepMind's effort at finding Scaling Laws

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*

"Compute optimal" : Best heldout cross-entropy given total FLOPs budget

- No constraints on # of GPUs and # Tokens
- Ignores communication latencies

$$N_{opt}(C), D_{opt}(C) = \underset{N,D \text{ s.t. } \text{FLOPs}(N,D)=C}{\text{argmin}} L(N, D).$$

Caveat: Minimization only over architectures, training, and datasets that were popular in '22

Experiments:

- 400 models, sizes 70M to 16B
- Dataset size 5B to 500B

(other hyper-parameters such as batch size, dimension, etc. taken from earlier studies)

"Compute optimal" : Best heldout cross-entropy given total FLOPs budget

- No constraints on # of GPUs and # Tokens
- Ignores communication latencies

$$N_{opt}(C), D_{opt}(C) = \underset{N,D \text{ s.t. } \text{FLOPs}(N,D)=C}{\text{argmin}} L(N, D).$$

Caveat: Minimization only over architectures, training, and datasets that were popular in '22

Let's figure out: If $L(N, D) = 2 + \frac{400}{N^{1/3}} + \frac{2000}{D^{1/3}}$ what is the correct scaling recipe?

Table: Scaling Recipe

Parameters	FLOPs	FLOPs (in *Gopher* unit)	Tokens
400 Million	1.92e+19	1/29,968	8.0 Billion
1 Billion	1.21e+20	1/4,761	20.2 Billion
10 Billion	1.23e+22	1/46	205.1 Billion
67 Billion	5.76e+23	1	1.5 Trillion
175 Billion	3.85e+24	6.7	3.7 Trillion
280 Billion	9.90e+24	17.2	5.9 Trillion
520 Billion	3.43e+25	59.5	11.0 Trillion
1 Trillion	1.27e+26	221.3	21.2 Trillion
10 Trillion	1.30e+28	22515.9	216.2 Trillion

"Chinchilla Scaling Law"
(D ≈ 20N is compute-optimal choice)

Main finding

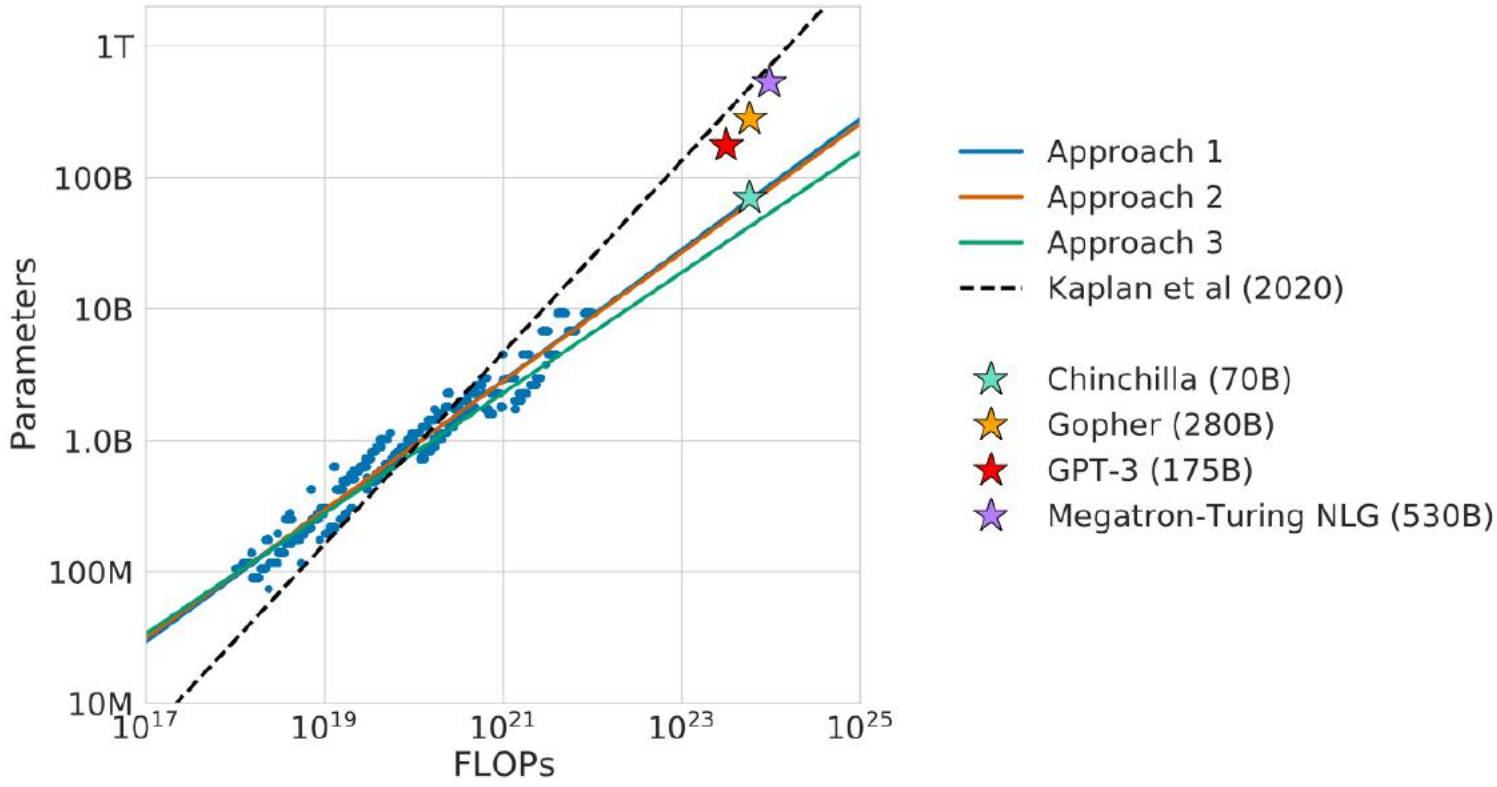


Figure 1 | Overlaid predictions. We overlay the predictions from our three different approaches, along with projections from Kaplan et al. (2020). We find that all three methods predict that current large models should be substantially smaller and therefore trained much longer than is currently done. In Figure A3, we show the results with the predicted optimal tokens plotted against the optimal number of parameters for fixed FLOP budgets. ***Chinchilla* outperforms *Gopher* and the other large models (see Section 4.2).**

Side-benefit: Compute optimal models => faster inference

Cosine LR schedule

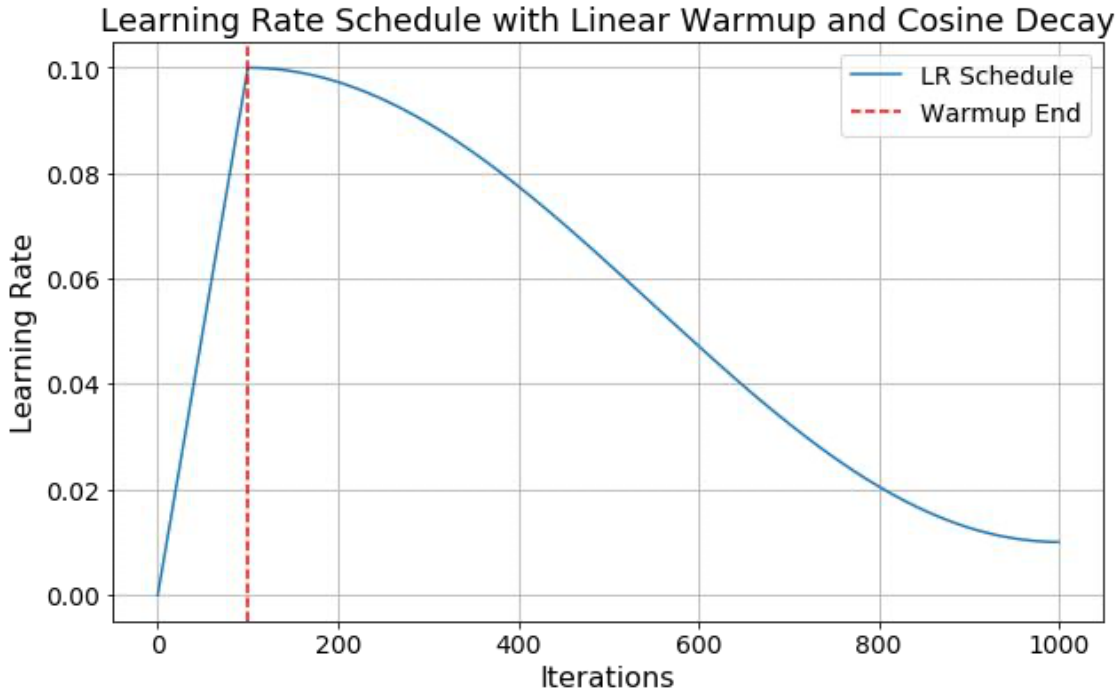
$$\mathrm{LR}(t) = \ell + \frac{1}{2}(L - \ell)\left[1 + \cos\left(\frac{(t - t_w)\pi}{(T - t_w)}\right)\right].$$

L = Max LR

ℓ = min LR

T = total # of iterations

t_w = # of warmup iterations



Key Finding: When # of tokens (hence T) changes, use LR schedule for this new T

(DON'T finish training before hitting the end of the cosine schedule.)

This partly explains why OpenAI's Scaling Law [Hoffman et al'20] was off..

Finding how loss scales with compute and data

IsoFlop Curves

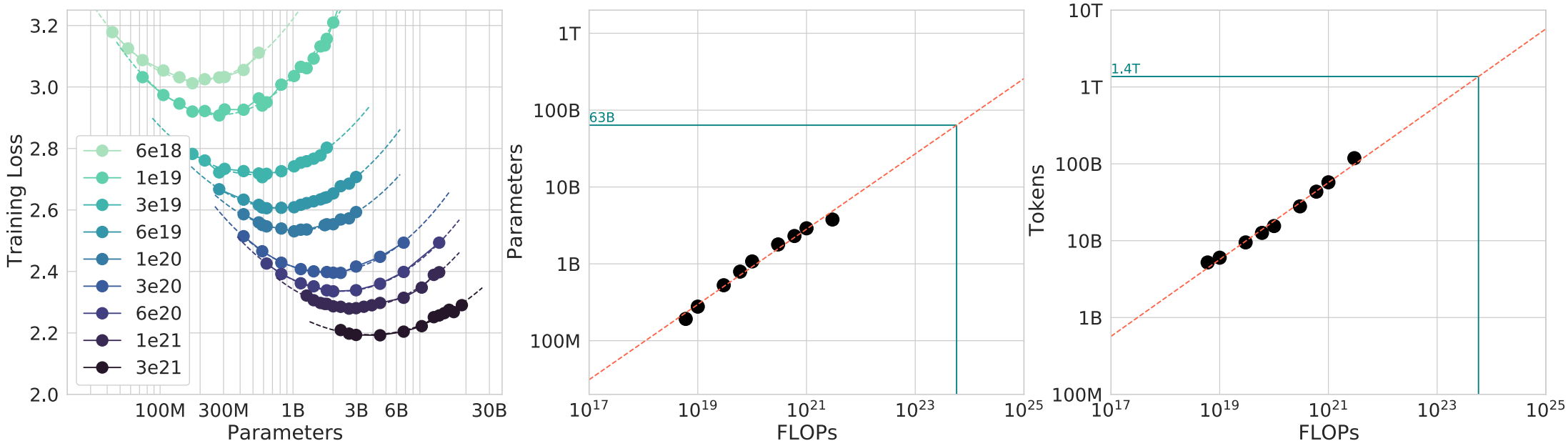


Figure 3 | **IsoFLOP curves.** For various model sizes, we choose the number of training tokens such that the final FLOPs is a constant. The cosine cycle length is set to match the target FLOP count. We find a clear valley in loss, meaning that for a given FLOP budget there is an optimal model to train (**left**). Using the location of these valleys, we project optimal model size and number of tokens for larger models (**center** and **right**). In green, we show the estimated number of parameters and tokens for an *optimal* model trained with the compute budget of *Gopher*.

Method to find Scaling Law

N = # parameters D = # tokens C = total compute

$$N_{opt}(C), D_{opt}(C) = \underset{N,D \text{ s.t. } \text{FLOPs}(N,D)=C}{\text{argmin}} L(N, D).$$

Empirical finding from prev. slide : $N = KC^{\alpha}, D = K^{-1}C^{\beta}$ for some α, β (functional form confirmed by all 3 approaches..)

Approach	Coeff. a where $N_{opt} \propto C^a$	Coeff. b where $D_{opt} \propto C^b$
1. Minimum over training curves	0.50 (0.488, 0.502)	0.50 (0.501, 0.512)
2. IsoFLOP profiles	0.49 (0.462, 0.534)	0.51 (0.483, 0.529)
3. Parametric modelling of the loss	0.46 (0.454, 0.455)	0.54 (0.542, 0.543)
Kaplan et al. (2020)	0.73	0.27

Estimated held-out c-e loss given D, N

$$L(N, D) = E + \frac{A}{N^{0.34}} + \frac{B}{D^{0.28}},$$

with $E = 1.69$, $A = 406.4$, $B = 410.7$.

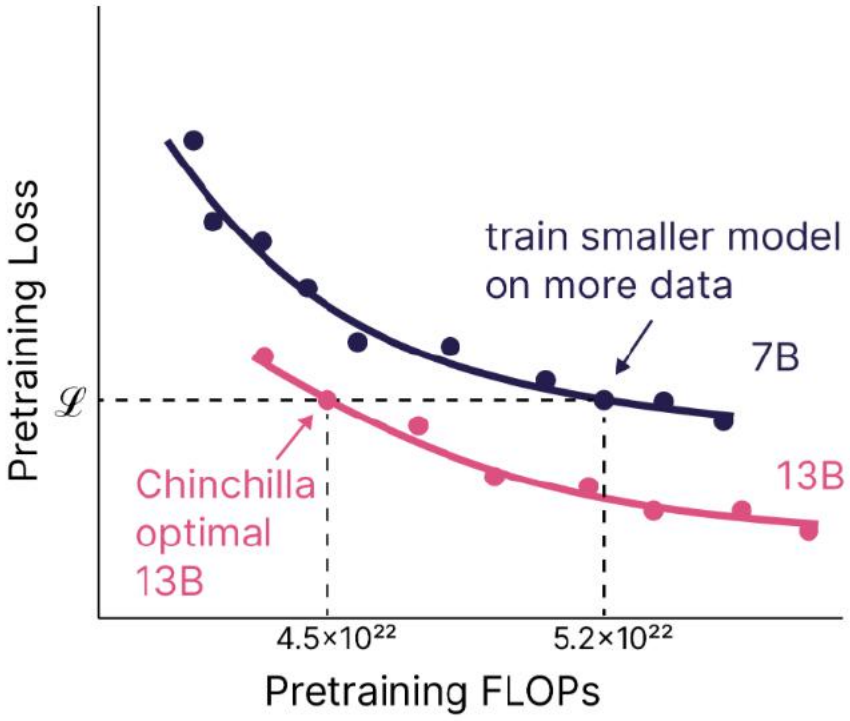
Fitting the decomposition to data. We effectively minimize the following problem

$$\min_{a,b,e,\alpha,\beta} \sum_{\text{Run } i} \text{Huber}_\delta\Big(\text{LSE}\big(a - \alpha \log N_i, b - \beta \log D_i, e\big) - \log L_i\Big),$$

where LSE is the log-sum-exp operator. We then set $A, B, E = \exp(a), \exp(b), \exp(e)$.

“Chinchilla Law”: Amended

v1: Accounting for inference cost



Depending on how many tokens are extracted in inference, higher cost of overtrained 7B model may be worth it

Beyond Chinchilla-Optimal:
Accounting for Inference in Language Model Scaling Laws

Nikhil Sardana [1] Jacob Portes [1] Sasha Doubov [1] Jonathan Frankle [1]

e.g., Llama1 7B trained on 1.4T tokens (Chinchilla recipe) in Feb'23, but a year later Llama3 8B was trained on 5T tokens

Correcting Mistakes in Chinchilla Paper

[Epoch AI, 2024]

Motivation: Accurate prediction on models that are not compute-optimal

We reconstruct a subset of the data in Hoffmann et al.'s paper by extracting it from their plots and fit the same parametric model. Our analysis reveals several potential issues with Hoffmann et al.'s estimates of the parameters of their scaling law:

1. Hoffmann et al.'s estimated model fits the reconstructed data poorly, even when accounting for potential noise in the data reconstruction and excluding outlier models.
2. The confidence intervals reported by Hoffmann et al. are implausibly tight given the likely number of data points they had (~400). Obtaining such tight intervals would require hundreds of thousands of observations.
3. The scaling policy implied by Hoffmann et al.'s estimated model is inconsistent with their other approaches and the 20-tokens-per-parameter rule of thumb used to train their Chinchilla model.

$$L(N, D) = 1.8172 + \frac{482.01}{N^{0.3478}} + \frac{2085.43}{D^{0.3658}}$$

Another major Chinchilla Amendation (data-constrained training)

Scaling Data-Constrained Language Models

Niklas Muennighoff [1] **Alexander M. Rush** [1] **Boaz Barak** [2] **Teven Le Scao** [1]
Aleksandra Piktus [1] **Nouamane Tazi** [1] **Sampo Pyysalo** [3] **Thomas Wolf** [1] **Colin Raffel** [1]

Motivation: Not enough data

e.g., Chinchilla law suggests training 530B model on 11T tokens

Assembling a dataset of 11T tokens may involve too many compromises (ie low-quality)

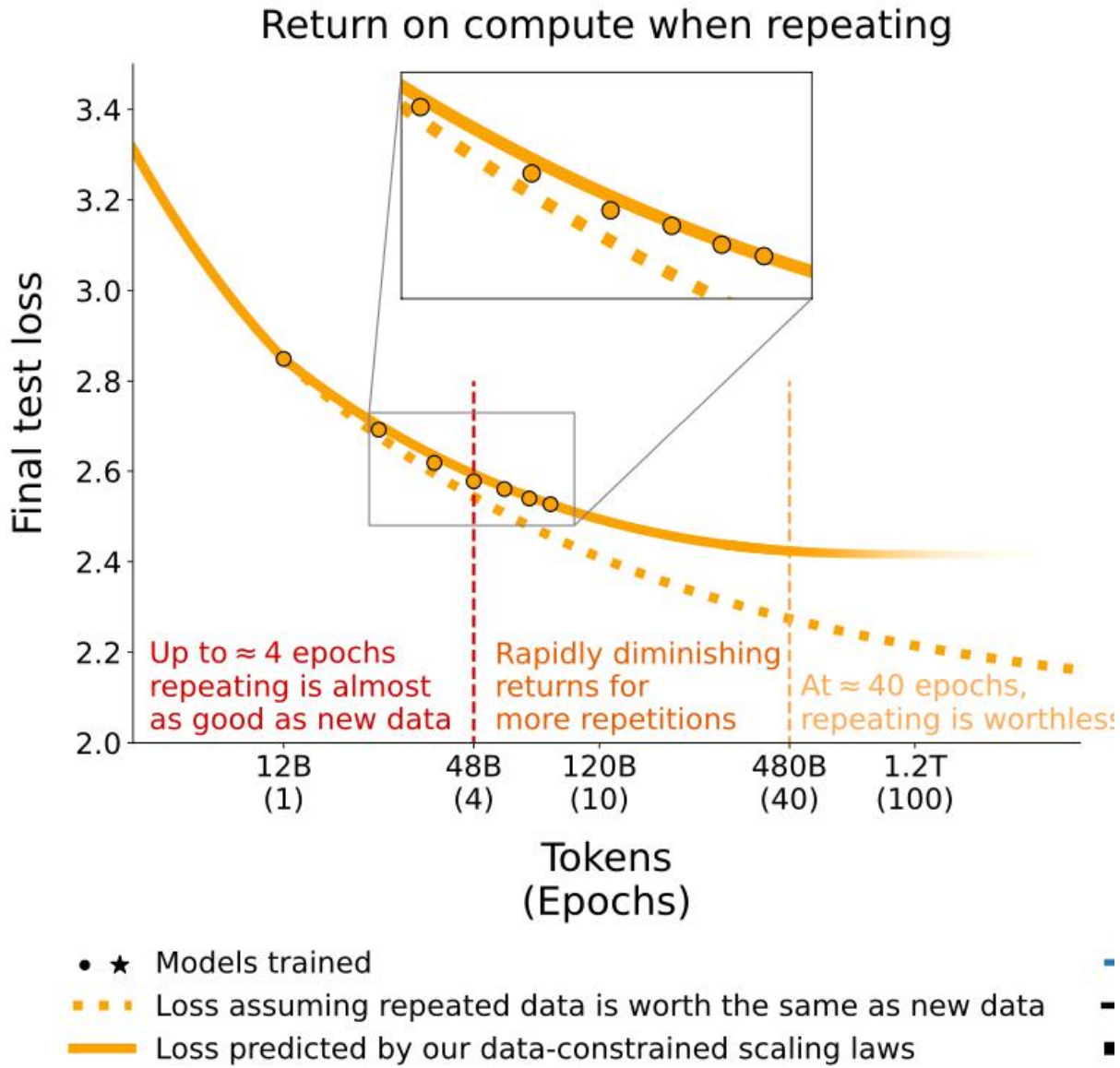
Specialized corpora (law, medicine, Wwikipedia etc.) are small; essentially fixed size.

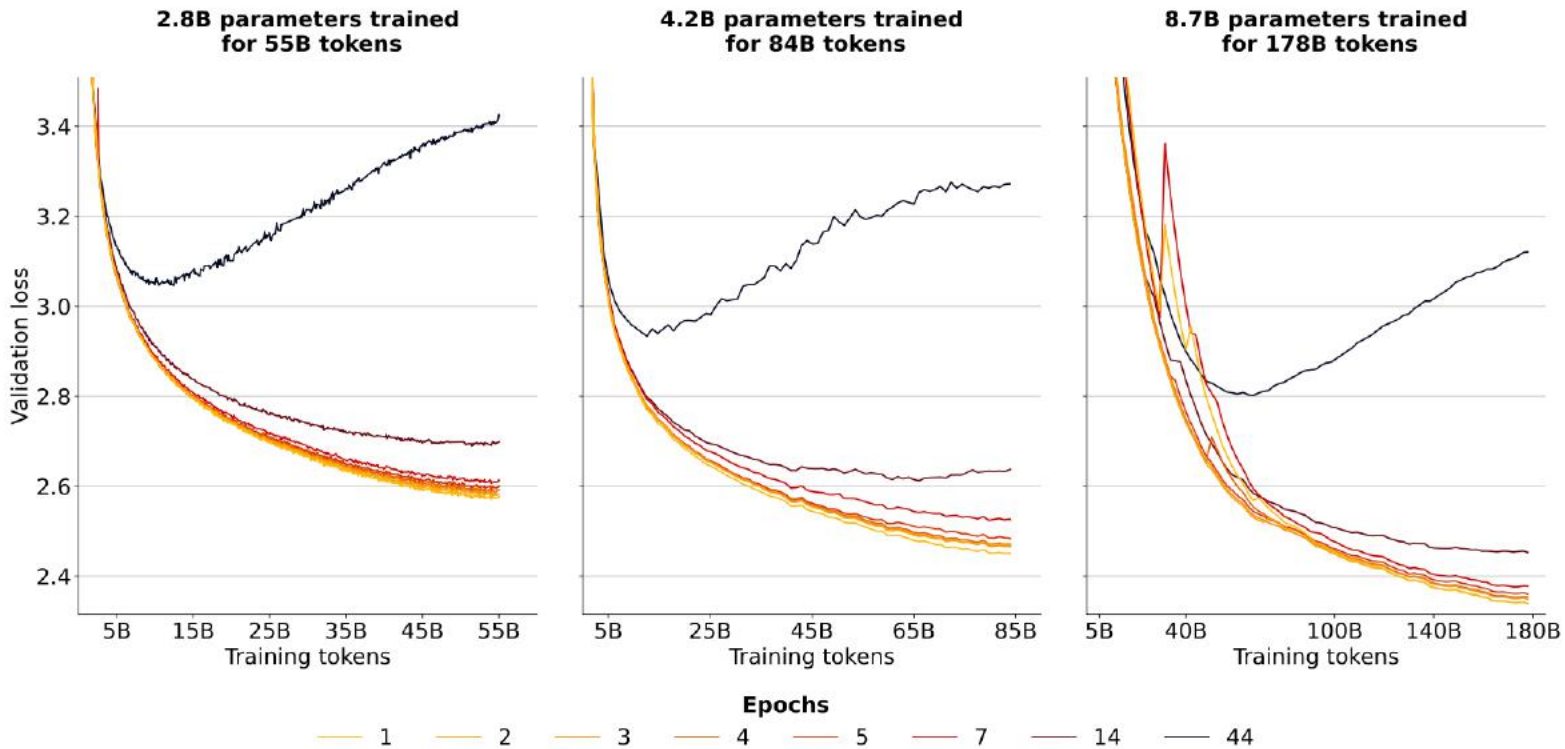
Scaling up with web data throws off the data-mix proportions

From paper abstract [Meunnighoff et al'23]

regimes. Specifically, we run a large set of experiments varying the extent of data repetition and compute budget, ranging up to 900 billion training tokens and 9 billion parameter models. We find that with constrained data for a fixed compute budget, training with up to 4 epochs of repeated data yields negligible changes to loss compared to having unique data. However, with more repetition, the value of adding compute eventually decays to zero. We propose and empirically validate a scaling law for compute optimality that accounts for the decreasing value of repeated tokens and excess parameters. Finally, we experiment with approaches mitigating data scarcity, including augmenting the training dataset with code data or removing commonly used filters. Models and datasets from our 400 training runs

Experiments with 4.2B model





FLOP budget (C)	Parameters (N)	Training tokens (D)	Data budget (D_C)
9.3×10^{20}	2.8B	55B	$\{55, 28, 18, 14, 11, 9, 4, 1.25\}$B
2.1×10^{21}	4.2B	84B	$\{84, 42, 28, 21, 17, 12, 6, 1.9\}$B
9.3×10^{21}	8.7B	178B	$\{178, 88, 58, 44, 35, 25, 13, 4\}$B

Figure 4: **Validation Loss for Different Data Constraints (IsoFLOP).** Each curve represents the same number of FLOPs spent on an equal size model. Colors represent different numbers of epochs due to repeating because of data constraints. Parameters and training tokens are set to match the single-epoch compute-optimal configurations for the given FLOPs. Models trained on data that is repeated for multiple epochs have consistently worse loss and diverge if too many epochs are used.

Thought process in deriving Chinchilla-like law

$$L(N, D) = \frac{A}{N'^{\alpha}} + \frac{B}{D'^{\beta}} + E$$

D' = "effective # of tokens"

N' = "effective # of parameters"

Let $D =$ total # of tokens with repetition. $U_D =$ unique tokens

Let $U_N =$ optimal # parameters for U_D tokens (as per Chinchilla)

Define $R_D = \frac{D}{U_D} - 1 \quad R_N = \frac{N}{U_N} - 1$

Hypothesis: There exist learnable parameters R_D^*, R_N^* such that

$D' =$ Effective datasize $= U_D + U_D R_D^* (1 - e^{-\frac{R_D}{R_D^*}})$

$N' = U_N + U_N R_N^* (1 - e^{\frac{-R_N}{R_N^*}})$.

Motivation: Exponential drop-off in effectiveness

Fitting this model

$$L(U_N, U_D, R_N, R_D) = \frac{A}{(U_N + U_N R_N^*(1 - e^{\frac{-R_N}{R_N^*}}))^\alpha} + \frac{B}{(U_D + U_D R_D^*(1 - e^{\frac{-R_D}{R_D^*}}))^\beta} + E$$

Do best fit using Huber loss $L_\delta(a) = \begin{cases} \frac{1}{2}a^2 & \text{if } |a| \le \delta \\ \delta(|a| - \frac{1}{2}\delta) & \text{if } |a| > \delta \end{cases}$

(Modification of MSE, less sensitive to outliers)

$R_N^* = 5.31, \quad R_D^* = 15.39$ fit the data quite well to give the expressionl

$$L(U_D, R_N, R_D) = \frac{521}{(U_N + 5.3 \cdot U_N(1 - e^{\frac{-R_N}{5.3}}))^{0.35}} + \frac{1488}{(U_D + 15.4 \cdot U_D(1 - e^{\frac{-R_D}{15.4}}))^{0.35}} + 1.87$$

where $U_N = U_D \cdot 0.051$

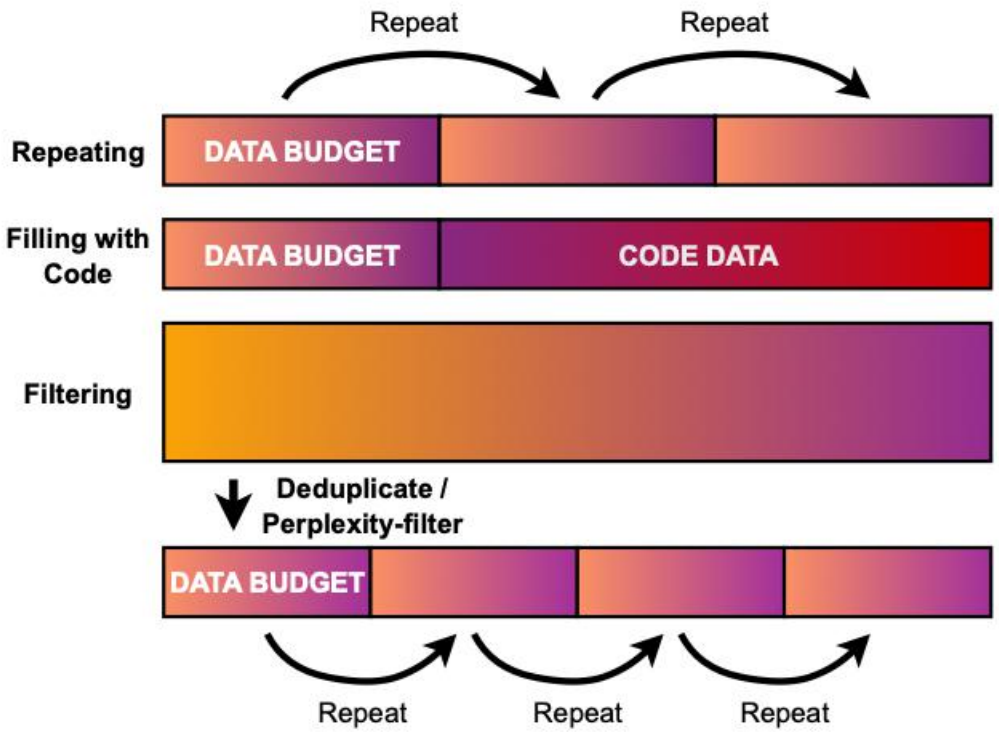
Two ways to overcome limited text data

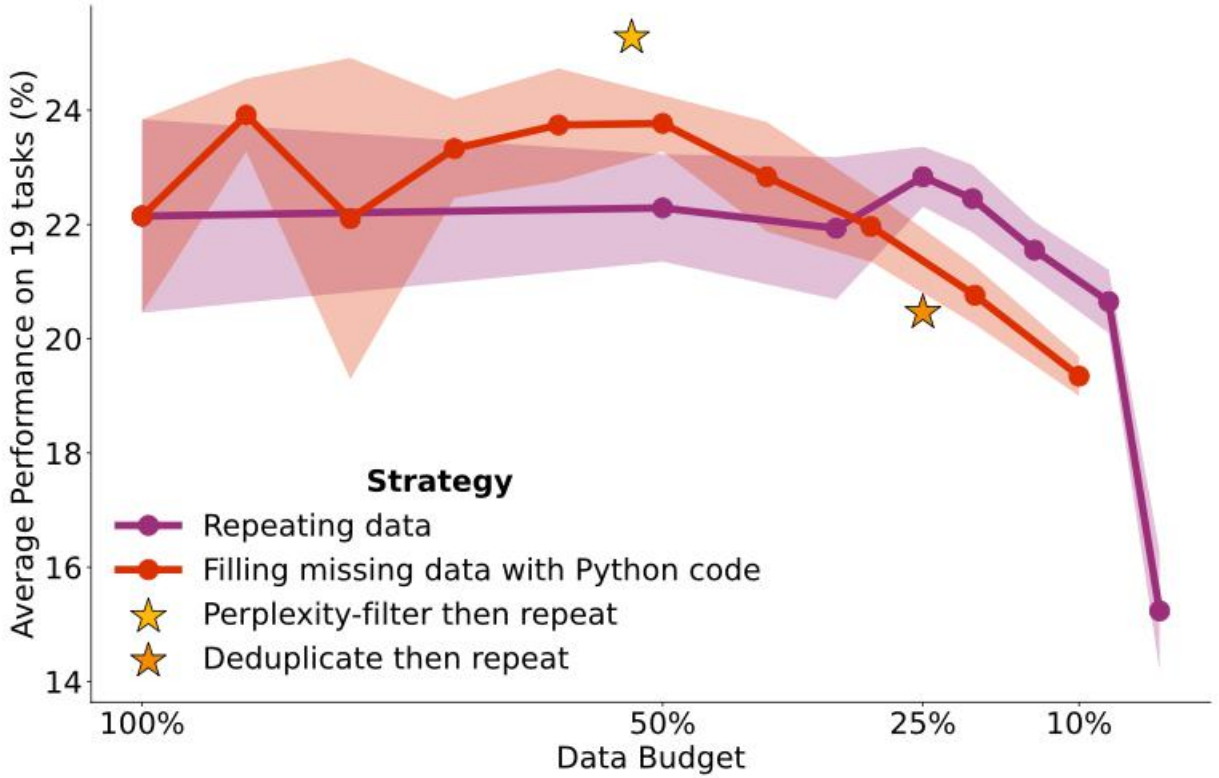
[Meunnighof et al.]

4.2B model, trained with 84B tokens. Tokens could be (i) unique (ii) repeated (iii) code tokens (iv) filtered using perplexity

For these experiments, we set a maximum data budget (D_C) of 84 billion tokens. For repetition and code filling, only a subset of D_C is available and the rest needs to be compensated for via repeating or adding code. For both filtering methods, we start out with approximately twice the budget (178 billion tokens), as it is easier to gather noisy data and filter it than it is to gather clean data for training. For perplexity filtering, we select the top 25% samples with the lowest perplexity according to a language model trained on Wikipedia. This results in 44 billion tokens that are repeated for close to two epochs to reach the full data budget. For deduplication filtering, all samples with a 100-char overlap are removed resulting in 21 billion tokens that are repeated for four epochs during training. See Appendix N for more details on the filtering procedures.

Note: "lowest perplexity" $\Longrightarrow$ highest probability (hopefully, "most like wikipedia")

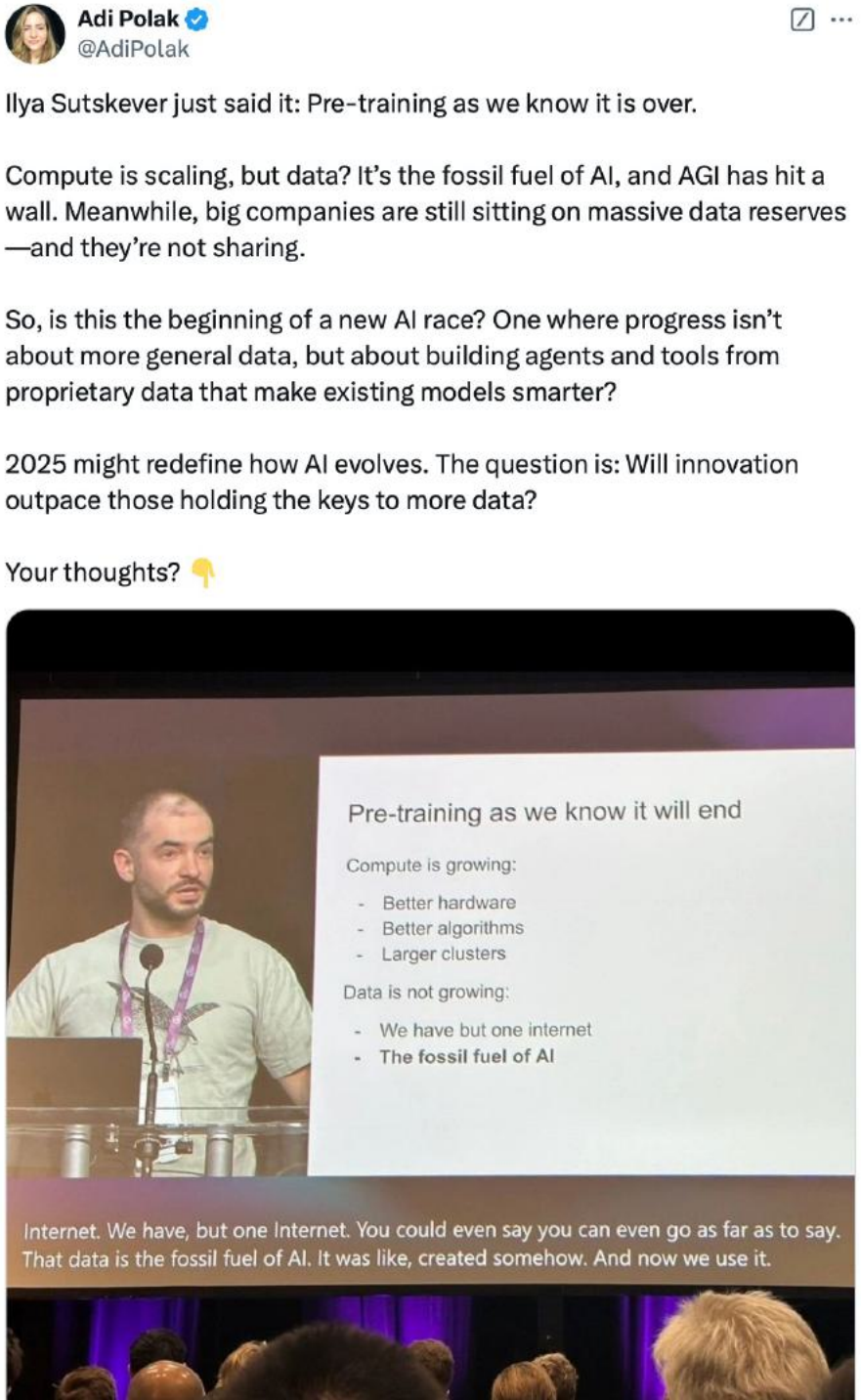




Interesting Settings: (i) Code + Data (up to 50-50 is good)
(ii) apply perplexity filter to get 42 B tokens, then 2 epochs

Caveat: Code is known to improve reasoning, and they didn't test for this

But...


Adi Polak
@AdiPolak
Ilya Sutskever just said it: Pre-training as we know it is over.
Compute is scaling, but data? It's the fossil fuel of AI, and AGI has hit a wall. Meanwhile, big companies are still sitting on massive data reserves —and they're not sharing.
So, is this the beginning of a new AI race? One where progress isn't about more general data, but about building agents and tools from proprietary data that make existing models smarter?
2025 might redefine how AI evolves. The question is: Will innovation outpace those holding the keys to more data?
Your thoughts? 👇
Pre-training as we know it will end
Compute is growing:
- Better hardware
- Better algorithms
- Larger clusters
Data is not growing:
- We have but one internet
- The fossil fuel of AI
Internet. We have, but one Internet. You could even say you can even go as far as to say. That data is the fossil fuel of AI. It was like, created somehow. And now we use it.

Llama 3

From GPT-3 to Llama 3

- GPT-1, GPT-2, GPT-3, GPT-3.5/ChatGPT, GPT-4, GPT-4-turbo, GPT-4o
- Llama 1, Llama 2, Llama 3
- Mistral, Mixtral
- Claude 1, Claude 2, Claude 3, Claude 3.5 (Haiku, Sonnet, Opus)
- Qwen 1, Qwen 2
- Bard, Gemini, Gemini Pro, Gemma 1, Gemma 2
- …
- Truly open LMs: OLMo, Pythia, BLOOM

More on these models later…

Llama 3.1: Overview

- **Dense Transformers** - 8B, 70B, 405B
  - Dense vs mixture-of-experts
  - Smaller models are getting more attention
- **Long-context:** 128K tokens (remember, GPT-3 had only 2048 tokens)
- **Pre-trained** on 15T multilingual tokens (remember, GPT-3 was trained on 300B tokens)
- Pre-training vs **post-training**:
  - SFT, rejection sampling, direct preference optimization
  - multilinguality, coding, reasoning, tool use
  - Safety mitigations: helpfulness vs harmlessness
- **Multi-modal** training and adaptation

Llama 3.1: Pre-training data

- "To train the best language model, the curation of a large, high-quality training dataset is paramount."
- PII and safety filtering
- Text extraction and cleaning from raw HTML pages
- De-duplication: URL, document, line-level, …
- **Heuristic filtering:**
  - Remove lines that consist of repeated content (e.g., n-gram coverage ratio)
  - Dirty word counting
  - KL divergence of token-distribution compared "high-quality corpus"
- **Model-based quality classifier:** important and new trend!
- **Code, reasoning, and multilingual** data

Llama 3.1: Heuristic filtering

C4 rules (Raffel et al., 2020)

- We only retained lines that ended in a terminal punctuation mark (i.e. a period, exclamation mark, question mark, or end quotation mark).
- We discarded any page with fewer than 5 sentences and only retained lines that contained at least 3 words.
- We removed any page that contained any word on the "List of Dirty, Naughty, Obscene or Otherwise Bad Words".[6]
- Many of the scraped pages contained warnings stating that Javascript should be enabled so we removed any line with the word Javascript.
- Some pages had placeholder "lorem ipsum" text; we removed any page where the phrase "lorem ipsum" appeared.
- Some pages inadvertently contained code. Since the curly bracket "{" appears in many programming languages (such as Javascript, widely used on the web) but not in natural text, we removed any pages that contained a curly bracket.

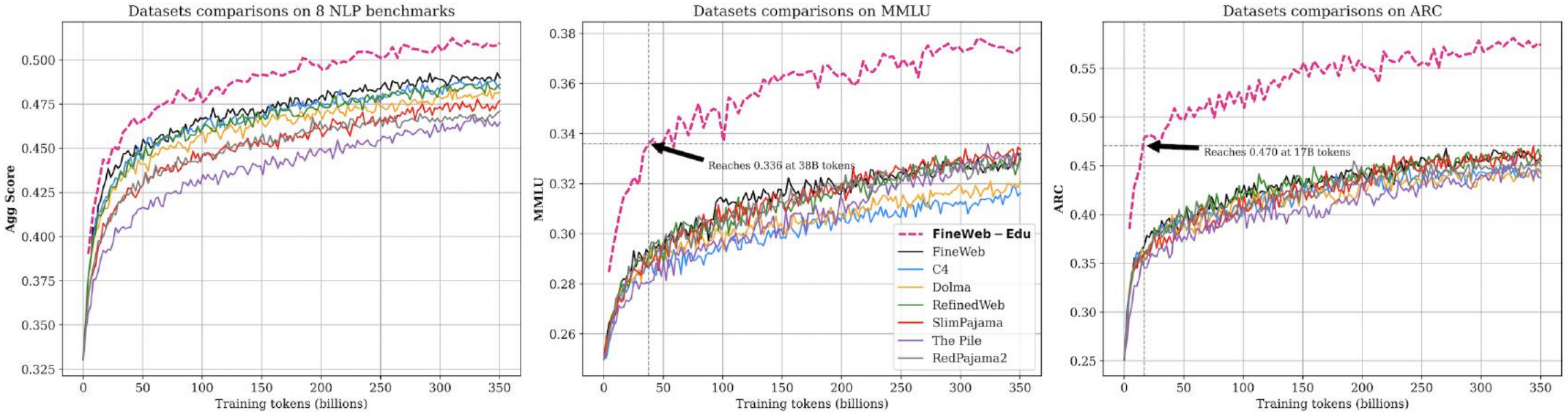
Gopher Rules (Rae et al., 2021)

```python
def gopher_rules_pass(sample) -> bool:
    """ function returns True if the sample complies with Gopher rules """
    signals = json.loads(sample["quality_signals"])

    # rule 1: number of words between 50 and 10'000
    word_count = signals["rps_doc_word_count"][0][2]
    if word_count < 50 or word_count > 10_000:
        return False

    # rule 2: mean word length between 3 and 10
    mean_word_length = signals["rps_doc_mean_word_length"][0][2]
    if mean_word_length < 3 or mean_word_length > 10:
        return False

    # rule 2: symbol to word ratio below 0.1
    symbol_word_ratio = signals["rps_doc_symbol_to_word_ratio"][0][2]
    if  symbol_word_ratio > 0.1:
        return False

    # rule 3: 90% of lines need to start without a bullet point
    n_lines = signals["ccnet_nlines"][0][2]
    n_lines_bulletpoint_start = sum(map(lambda ln: ln[2], signals["rps_lines_start_w
    if n_lines_bulletpoint_start / n_lines > 0.9:
        return False

    # rule 4: the ratio between characters in the most frequent 2-gram and the total
    # of characters must be below 0.2
    top_2_gram_frac = signals["rps_doc_frac_chars_top_2gram"][0][2]
    if top_2_gram_frac > 0.2:
        return False

    # rule 5: ...
```

Llama 3.1: Model-based quality filtering

"To train a quality classifier based on **Llama 2**, we create a training set of cleaned web documents, **describe the quality requirements**, and **instruct Llama 2's chat model to determine if the documents meets these requirements**. We use **DistilRoberta** (Sanh et al., 2019) to generate quality scores for each document for efficiency reasons. We experimentally evaluate the efficacy of various quality filtering configurations."

FINEWEB-EDU They generate **450k annotations** by **llama-3-instruct** for identifying educational content



Llama 3.1: Model-based quality filtering

Below is an extract from a web page. Evaluate whether the page has a high educational value and could be useful in an educational setting for teaching from primary school to grade school levels using the additive 5-point scoring system described below. Points are accumulated based on the satisfaction of each criterion:

- Add 1 point if the extract provides some basic information relevant to educational topics, even if it includes some irrelevant or non-academic content like advertisements and promotional material.
- Add another point if the extract addresses certain elements pertinent to education but does not align closely with educational standards. It might mix educational content with non-educational material, offering a superficial overview of potentially useful topics, or presenting information in a disorganized manner and incoherent writing style.
- Award a third point if the extract is appropriate for educational use and introduces key concepts relevant to school curricula. It is coherent though it may not be comprehensive or could include some extraneous information. It may resemble an introductory section of a textbook or a basic tutorial that is suitable for learning but has notable limitations like treating concepts that are too complex for grade school students.
- Grant a fourth point if the extract highly relevant and beneficial for educational purposes for a level not higher than grade school, exhibiting a clear and consistent writing style. It could be similar to a chapter from a textbook or a tutorial, offering substantial educational content, including exercises and solutions, with minimal irrelevant information, and the concepts aren't too advanced for grade school students. The content is coherent, focused, and valuable for structured learning.
- Bestow a fifth point if the extract is outstanding in its educational value, perfectly suited for teaching either at primary school or grade school. It follows detailed reasoning, the writing style is easy to follow and offers profound and thorough insights into the subject matter, devoid of any non-educational or complex content.

The extract: <extract>.

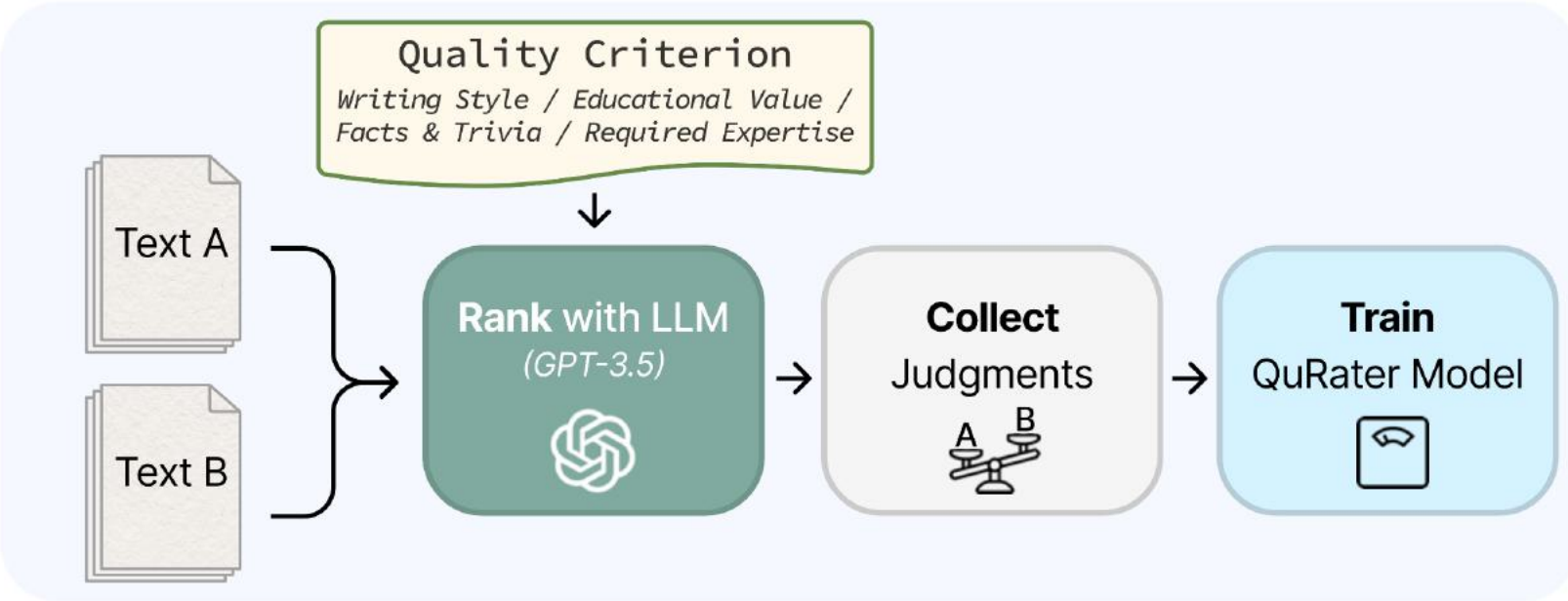
After examining the extract:

- Briefly justify your total score, up to 100 words.
- Conclude with the score using the format: "Educational score: <total points>"

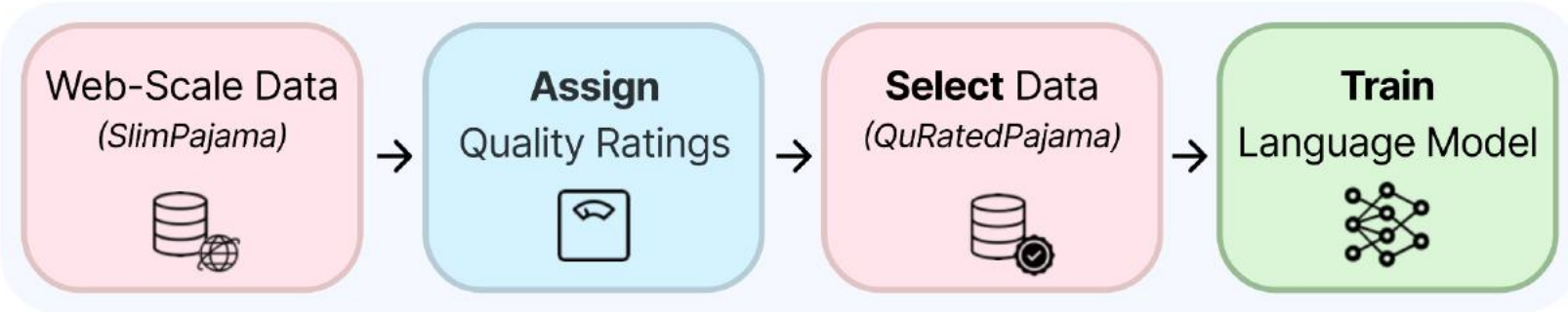
https://huggingface.co/datasets/HuggingFaceFW/fineweb-edu

Selecting high-quality data with LM signals

Part I
measure quality

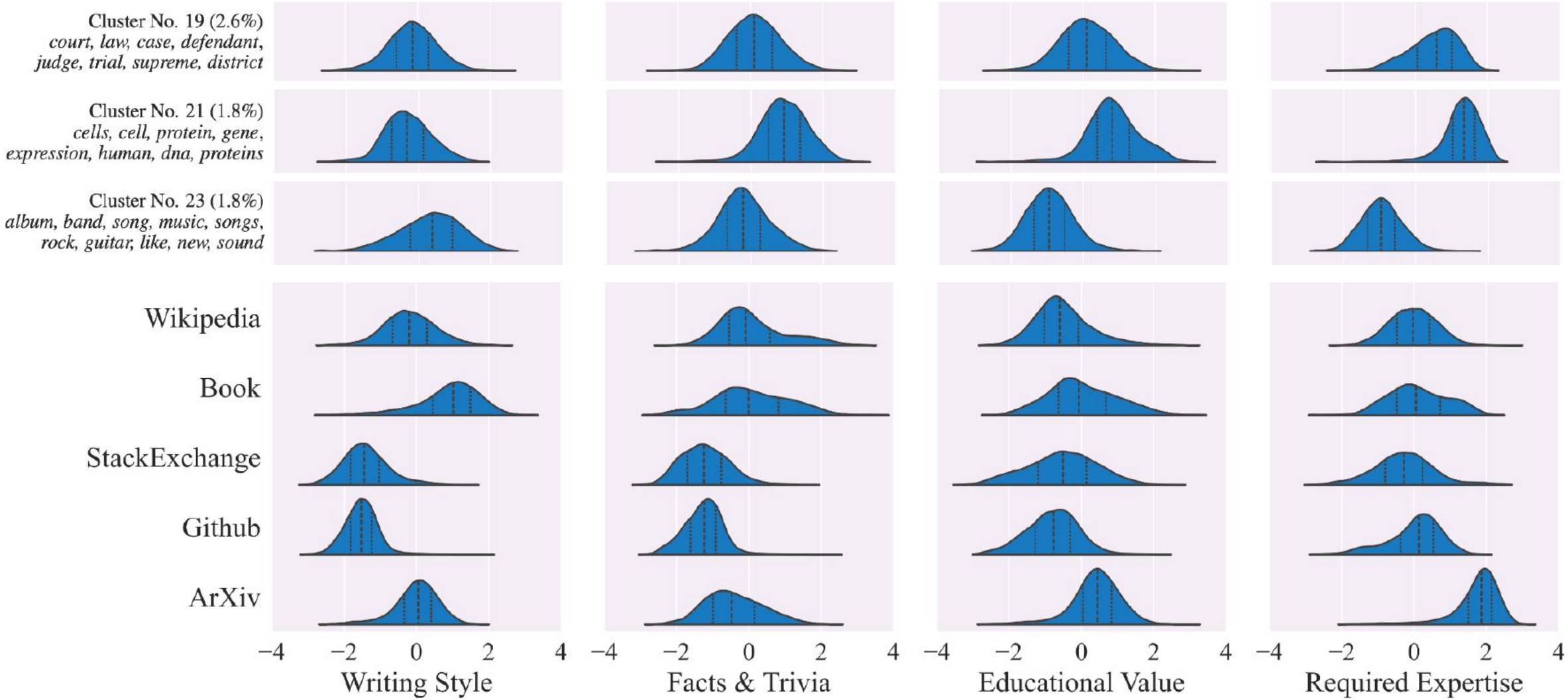


Part II
utilize quality



QuRating: Selecting High-Quality Data for Training Language Models (2024)

Selecting high-quality data with LM signals



QuRating: Selecting High-Quality Data for Training Language Models (2024)

Code and math data

- Common wisdom: code and math data are very important for pre-training
- They build domain-specific pipelines that extract code and math-relevant web pages

Published as a conference paper at ICLR 2024

AT WHICH TRAINING STAGE DOES CODE DATA HELP LLMS REASONING?

Yingwei Ma[1,2*], **Yue Liu**[1*], **Yue Yu**[1,2†], **Yuanliang Zhang**[1], **Yu Jiang**[3], **Changjian Wang**[1], **Shanshan Li**[1†]
[1]National University of Defense Technology
[2]Peng Cheng Laboratory
[3]Tsinghua University

To Code, or Not To Code?
Exploring Impact of Code in Pre-training

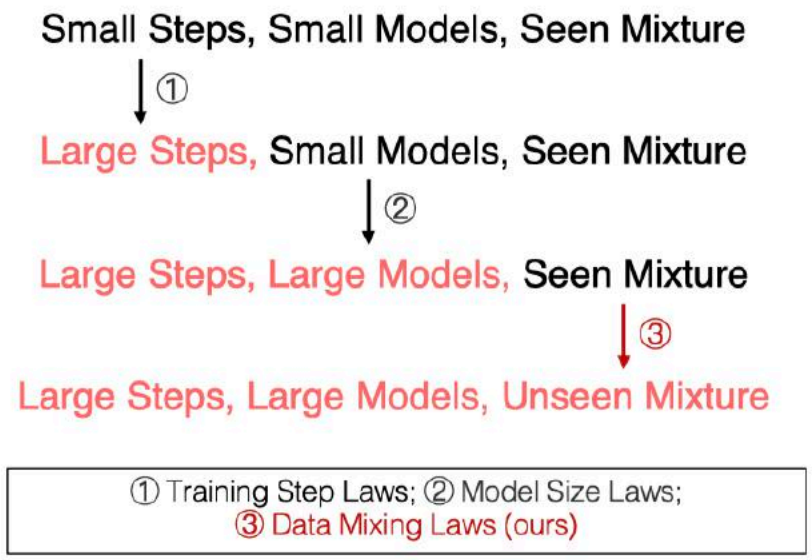
Viraat Aryabumi[1], **Yixuan Su**[2], **Raymond Ma**[2], **Adrien Morisot**[2], **Ivan Zhang**[2], **Acyr Locatelli**[2], **Marzieh Fadaee**[1], **Ahmet Üstün**[1], **and Sara Hooker**[1]

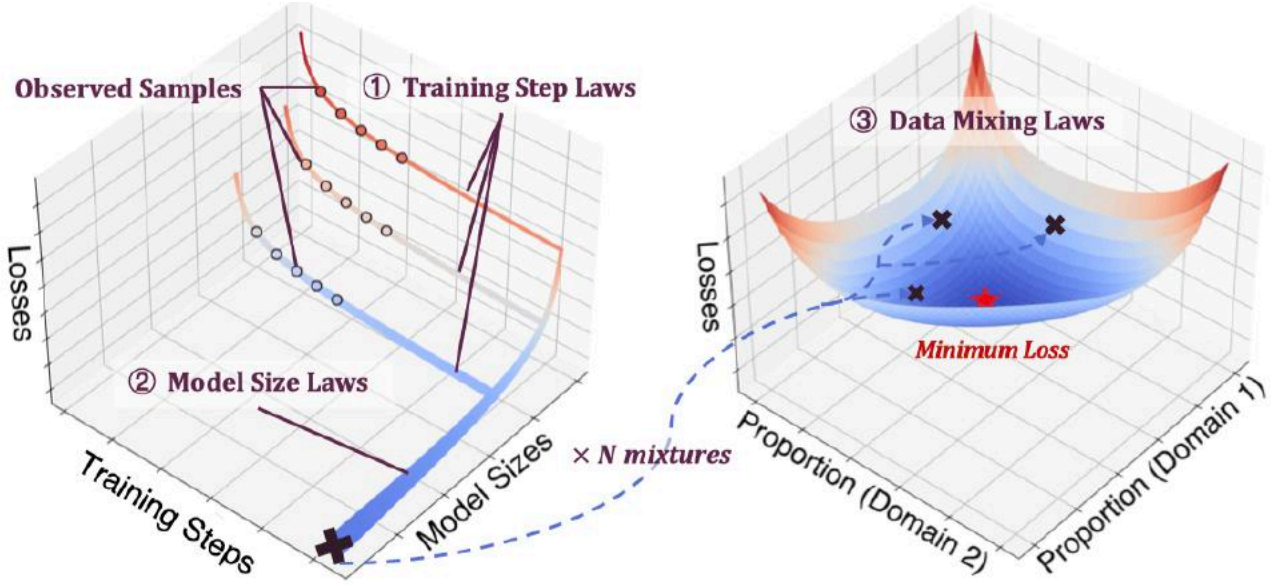
[1]Cohere For AI, [2]Cohere

- Code is a critical building block for generalization far beyond coding tasks
  - Compared to text-only pre-training, 8.2% in NL reasoning, 4.2% in world knowledge, 6.6% in general win rates, 12x in code performance
- The quality of code data has an outsized impact in downstream tasks

Determining data mix

- "Roughly **50%** of tokens corresponding to general knowledge, **25%** of mathematical and reasoning tokens, **17%** code tokens, **8%** multilingual tokens"
- **Scaling laws for data mix:** "train several smaller models on a data mix and use that to predict the performance on that mix", "repeat this process for different data mixes to select a new data mix candidate"

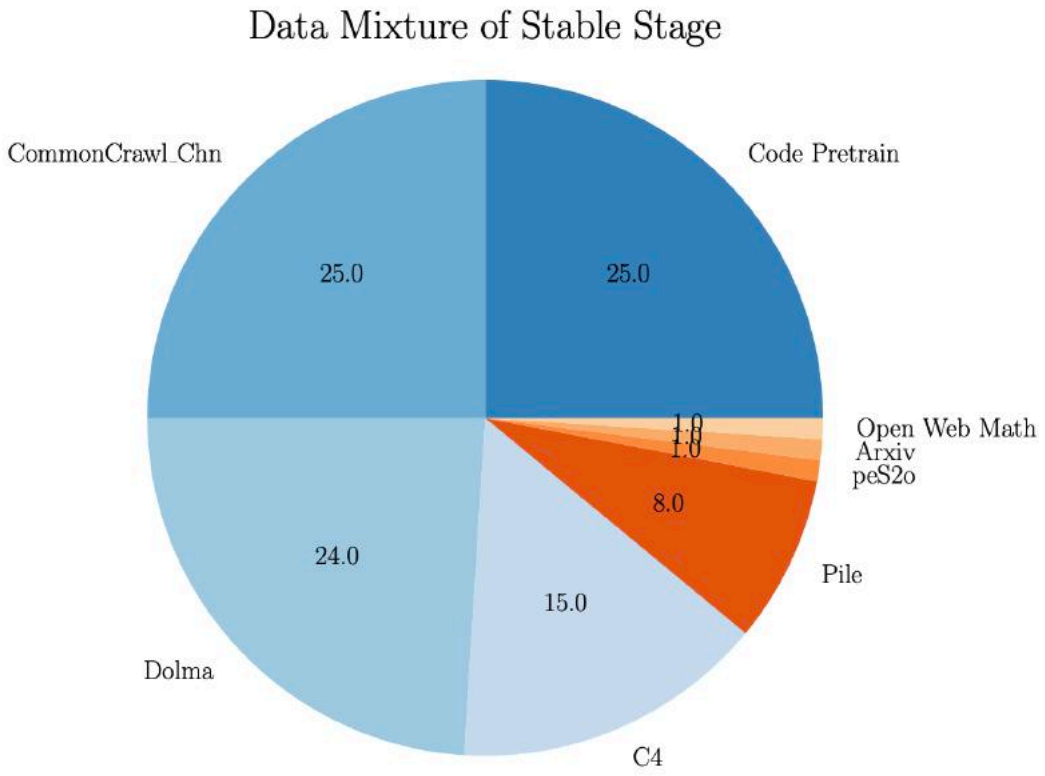


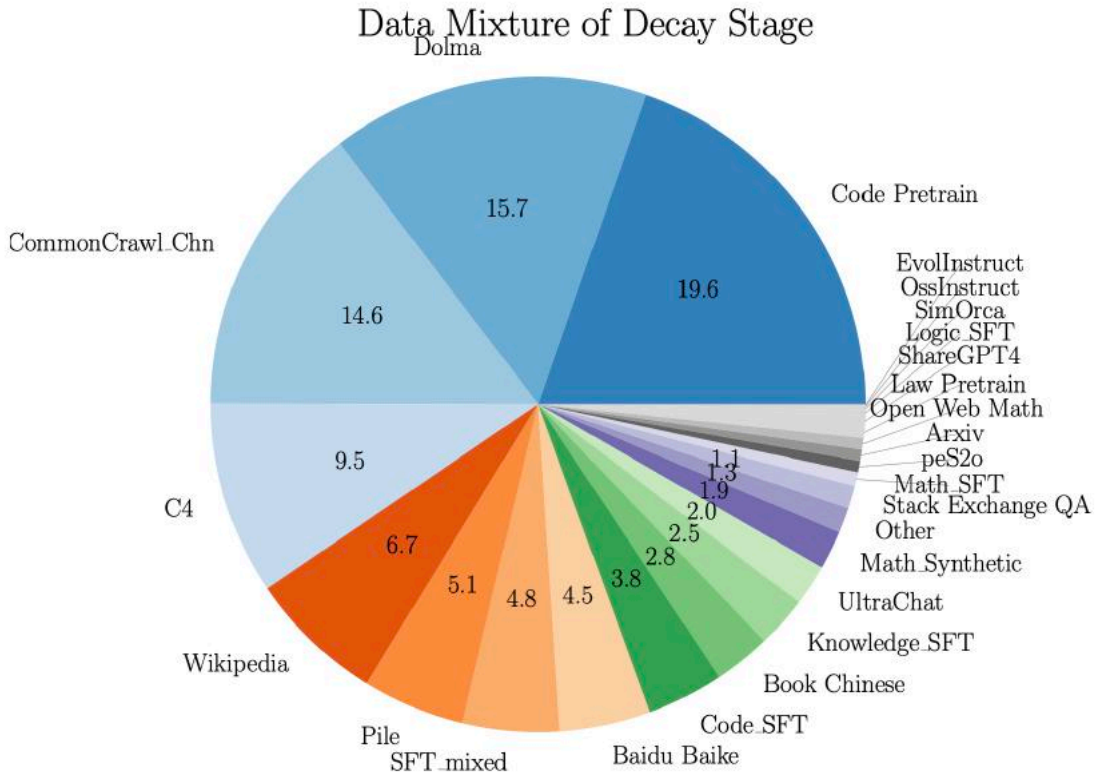


Data Mixing Laws: Optimizing Data Mixtures by Predicting Language Modeling Performance (2024)

Determining data mix

- **Domains:** Common Crawl, CC, Github, Wikipedia, Books, arXiv, …

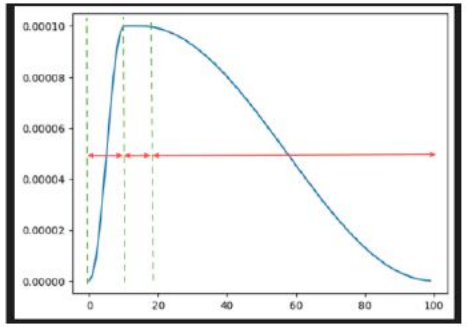




MiniCPM: Unveiling the Potential of Small Language Models with Scalable Training Strategies (2024)

Training recipe

- AdamW: learning rate of 8×10^{-5}, a linear warm up of 8000 steps, and a cosine learning rate schedule decaying to 8×10^{-7} over 1,200,000 steps



Cosine LR schedule with linear warmup

- They adjusted the pre-training mix during training
  - Increased percentage of non-English data
  - Upsample mathematical data to improve the model's knowledge cut-off
  - Downsampled subsets of pre-training data that were later identified as lower quality
- **Long-context pre-training:** first train on 8k, and increase context length to 128k in six stages (800B training tokens)
  - Challenges: scarcity of real long-context pre-training data
  - The performance on short-context tasks will degrade drastically

Data annealing

- They upsample on data sources of very high-quality at the end of training (final 40M tokens; no benchmark datasets used in annealing)

Does your data spark joy? Performance gains from domain upsampling at the end of training

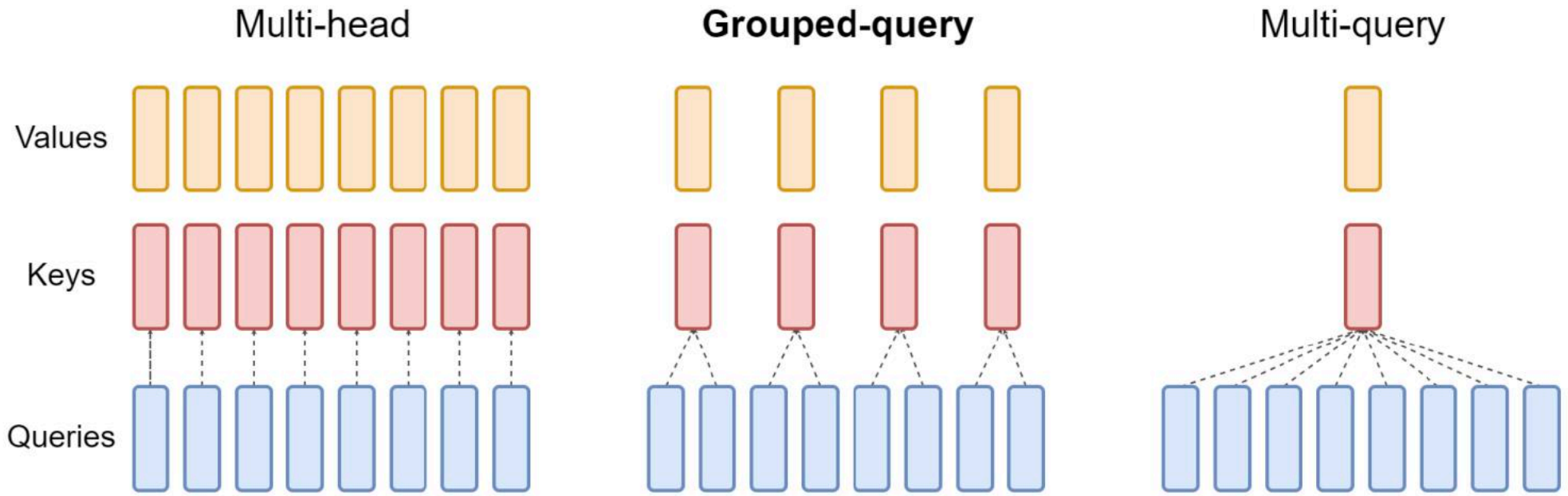
Cody Blakeney*, Mansheej Paul*, Brett W. Larsen*, Sean Owen, and Jonathan Frankle

Databricks Mosaic Research

- They view data annealing as a cheap way to measure the impact of domain-specific datasets on model capabilities

Model architecture

- Standard dense Transformers, the same architecture as Llama-2
- **Grouped query attention (GQA):** 8 key-value heads to improve inference speed

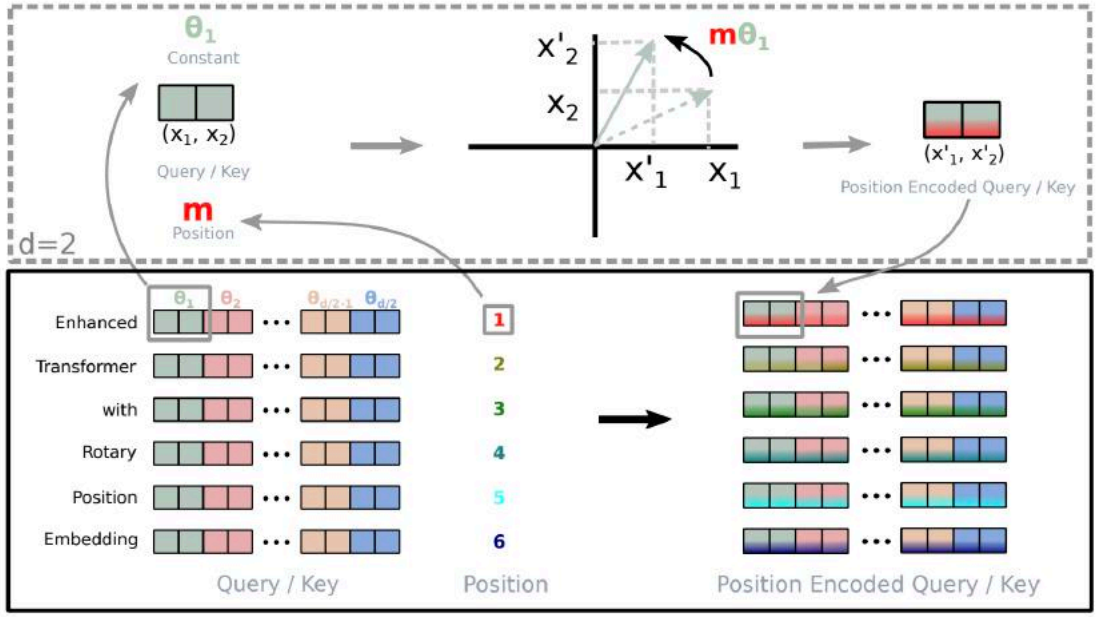


(Ainslie et al., 2023) GQA: Training generalized multi-query transformer models from multi-head checkpoints.

Model architecture

- Standard dense Transformers, the same architecture as Llama-2
- **Grouped query attention (GQA):** 8 key-value heads to improve inference speed
- Prevents self-attention between documents within the same sequence
- A much larger vocabulary: 128K
- **RoPE positional embeddings:** base frequency = 500,000

Rope positional embeddings



$$f_{\{q,k\}}(\boldsymbol{x}_m, m) = \begin{pmatrix} \cos m\theta & -\sin m\theta \\ \sin m\theta & \cos m\theta \end{pmatrix} \begin{pmatrix} W^{(11)}_{\{q,k\}} & W^{(12)}_{\{q,k\}} \\ W^{(21)}_{\{q,k\}} & W^{(22)}_{\{q,k\}} \end{pmatrix} \begin{pmatrix} x^{(1)}_m \\ x^{(2)}_m \end{pmatrix}$$

$$f_{\{q,k\}}(\boldsymbol{x}_m, m) = \boldsymbol{R}^d_{\Theta,m} \boldsymbol{W}_{\{q,k\}} \boldsymbol{x}_m$$

where

$$\boldsymbol{R}^d_{\Theta,m} = \begin{pmatrix} \cos m\theta_1 & -\sin m\theta_1 & 0 & 0 & \cdots & 0 & 0 \\ \sin m\theta_1 & \cos m\theta_1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cos m\theta_2 & -\sin m\theta_2 & \cdots & 0 & 0 \\ 0 & 0 & \sin m\theta_2 & \cos m\theta_2 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \cos m\theta_{d/2} & -\sin m\theta_{d/2} \\ 0 & 0 & 0 & 0 & \cdots & \sin m\theta_{d/2} & \cos m\theta_{d/2} \end{pmatrix}$$

$$\theta_i = 10000^{-2(i-1)/d}, i \in [1, 2, ..., d/2]$$

Base frequency

(Su et al., 2021) RoFormer: Enhanced Transformer with Rotary Position Embedding

Evaluation

Reading Comprehension	SQuAD V2 (Rajpurkar et al., 2018), QuaC (Choi et al., 2018), RACE (Lai et al., 2017),
Code	HumanEval (Chen et al., 2021), MBPP (Austin et al., 2021),
Commonsense reasoning/understanding	CommonSenseQA (Talmor et al., 2019), PiQA (Bisk et al., 2020), SiQA (Sap et al., 2019), OpenBookQA (Mihaylov et al., 2018), WinoGrande (Sakaguchi et al., 2021)
Math, reasoning, and problem solving	GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021b), ARC Challenge (Clark et al., 2018), DROP (Dua et al., 2019), WorldSense (Benchekroun et al., 2023)
Adversarial	Adv SQuAD (Jia and Liang, 2017), Dynabench SQuAD (Kiela et al., 2021), GSM-Plus (Li et al., 2024c) PAWS (Zhang et al., 2019)
Long context	QuALITY (Pang et al., 2022), many-shot GSM8K (An et al., 2023a)
Aggregate	MMLU (Hendrycks et al., 2021a), MMLU-Pro (Wang et al., 2024b), AGIEval (Zhong et al., 2023), BIG-Bench Hard (Suzgun et al., 2023)

Performance: Reading comprehension

	Reading Comprehension		
	SQuAD	QuAC	RACE
Llama 3 8B	77.0 ±0.8	**44.9** ±1.1	**54.3** ±1.4
Mistral 7B	73.2 ±0.8	44.7 ±1.1	53.0 ±1.4
Gemma 7B	**81.8** ±0.7	42.4 ±1.1	48.8 ±1.4
Llama 3 70B	81.8 ±0.7	**51.1** ±1.1	59.0 ±1.4
Mixtral 8×22B	**84.1** ±0.7	44.9 ±1.1	**59.2** ±1.4
Llama 3 405B	**81.8** ±0.7	**53.6** ±1.1	**58.1** ±1.4
GPT-4	–	–	–
Nemotron 4 340B	–	–	–
Gemini Ultra	–	–	–

	Math and Reasoning	
	ARC-C	DROP
Llama 3 8B	**79.7** ±2.3	**59.5** ±1.0
Mistral 7B	78.2 ±2.4	53.0 ±1.0
Gemma 7B	78.6 ±2.4	56.3 ±1.0
Llama 3 70B	**92.9** ±1.5	**79.6** ±0.8
Mixtral 8×22B	91.9 ±1.6	77.5 ±0.8
Llama 3 405B	96.1 ±1.1	**84.8** ±0.7
GPT-4	**96.3** ±1.1	80.9 ±0.8
Nemotron 4 340B	94.3 ±1.3	–
Gemini Ultra	–	$82.4^{\triangle}$ ±0.8

DROP: 3-shot, SQuAD: 1-shot, RACE: 0-shot, QuAC: 1-shot, ARC-C: 25-shot..

Setting	CoQA	DROP	QuAC	SQuADv2	RACE-h	RACE-m
Fine-tuned SOTA	**90.7**[a]	**89.1**[b]	**74.4**[c]	**93.0**[d]	**90.0**[e]	**93.1**[e]
GPT-3 Zero-Shot	81.5	23.6	41.5	59.5	45.5	58.4
GPT-3 One-Shot	84.0	34.3	43.3	65.4	45.9	57.4
GPT-3 Few-Shot	85.0	36.5	44.3	69.8	46.8	58.1

Setting	ARC (Challenge)
Fine-tuned SOTA	**78.5**[KKS+20]
GPT-3 Zero-Shot	51.4
GPT-3 One-Shot	53.2
GPT-3 Few-Shot	51.5

Performance: Commonsense reasoning

	Commonsense Understanding		
	PiQA	OpenBookQA	Winogrande
Llama 3 8B	81.0 ±1.8	45.0 ±4.4	75.7 ±2.0
Mistral 7B	**83.0** ±1.7	47.8 ±4.4	**78.1** ±1.9
Gemma 7B	81.5 ±1.8	**52.8** ±4.4	74.7 ±2.0
Llama 3 70B	83.8 ±1.7	47.6 ±4.4	83.5 ±1.7
Mixtral 8×22B	**85.5** ±1.6	**50.8** ±4.4	**84.7** ±1.7
Llama 3 405B	**85.6** ±1.6	**49.2** ±4.4	82.2 ±1.8
GPT-4	–	–	87.5 ±1.5
Nemotron 4 340B	–	–	**89.5** ±1.4

PiQA: 0-shot, OpenBookQA: 0-shot, Winogrande: 5-shot

Setting	PIQA	OpenBookQA
Fine-tuned SOTA	79.4	**87.2**[KKS+20]
GPT-3 Zero-Shot	**80.5***	57.6
GPT-3 One-Shot	**80.5***	58.8
GPT-3 Few-Shot	**82.8***	65.4

Setting	Winogrande (XL)
Fine-tuned SOTA	**84.6**[b]
GPT-3 Zero-Shot	70.2
GPT-3 One-Shot	73.2
GPT-3 Few-Shot	77.7

Performance: Code and math

HUMANEVAL

```python
def incr_list(l: list):
    """Return list with elements incremented by 1.
    >>> incr_list([1, 2, 3])
    [2, 3, 4]
    >>> incr_list([5, 3, 5, 2, 3, 3, 9, 0, 123])
    [6, 4, 6, 3, 4, 4, 10, 1, 124]
    """
    return [i + 1 for i in l]
```

```python
def solution(lst):
    """Given a non-empty list of integers, return the sum of all of the odd elements
    that are in even positions.

    Examples
    solution([5, 8, 7, 1]) ==>12
    solution([3, 3, 3, 3, 3]) ==>9
    solution([30, 13, 24, 321]) ==>0
    """
    return sum(lst[i] for i in range(0,len(lst)) if i % 2 == 0 and lst[i] % 2 == 1)
```

```python
def encode_cyclic(s: str):
    """
    returns encoded string by cycling groups of three characters.
    """
    # split string to groups. Each of length 3.
    groups = [s[(3 * i):min((3 * i + 3), len(s))] for i in range((len(s) + 2) // 3)]
    # cycle elements in each group. Unless group has fewer elements than 3.
    groups = [(group[1:] + group[0]) if len(group) == 3 else group for group in groups]
    return "".join(groups)

def decode_cyclic(s: str):
    """
    takes as input string encoded with encode_cyclic function. Returns decoded string.
    """
    # split string to groups. Each of length 3.
    groups = [s[(3 * i):min((3 * i + 3), len(s))] for i in range((len(s) + 2) // 3)]
    # cycle elements in each group.
    groups = [(group[-1] + group[:-1]) if len(group) == 3 else group for group in groups]
    return "".join(groups)
```

GSM8K

Problem

The battery charge in Mary's cordless vacuum cleaner lasts ten minutes. It takes her four minutes to vacuum each room in her house. Mary has three bedrooms, a kitchen, and a living room. How many times does Mary need to charge her vacuum cleaner to vacuum her whole house?

Solution

Mary has 3 + 1 + 1 = 5 rooms in her house.
At 4 minutes a room, it will take her 4 * 5 = 20 minutes to vacuum her whole house.
At 10 minutes a charge, she will need to charge her vacuum cleaner 20 / 10 = 2 times to vacuum her whole house.

Final Answer

2

Performance: Code and math

	Code	
	HumanEval	MBPP
Llama 3 8B	**37.2** ±7.4	**47.6** ±4.4
Mistral 7B	30.5 ±7.0	47.5 ±4.4
Gemma 7B	32.3 ±7.2	44.4 ±4.4
Llama 3 70B	**58.5** ±7.5	66.2 ±4.1
Mixtral 8×22B	45.1 ±7.6	**71.2** ±4.0
Llama 3 405B	61.0 ±7.5	**73.4** ±3.9
GPT-4	67.0 ±7.2	–
Nemotron 4 340B	57.3 ±7.6	–
Gemini Ultra	**74.4** ±6.7	–

	GSM8K	MATH
Llama 3 8B	**57.2** ±2.7	20.3 ±1.1
Mistral 7B	52.5 ±2.7	13.1 ±0.9
Gemma 7B	46.4 ±2.7	**24.3** ±1.2
Llama 3 70B	83.7 ±2.0	41.4 ±1.4
Mixtral 8×22B	**88.4** ±1.7	**41.8** ±1.4
Llama 3 405B	89.0 ±1.7	**53.8** ±1.4
GPT-4	**92.0** ±1.5	–
Nemotron 4 340B	–	–
Gemini Ultra	88.9$^{\diamond}$ ±1.7	53.2±1.4

Contamination analysis

	Contam.	Performance gain est.		
		8B	70B	405B
AGIEval	98	8.5	19.9	16.3
BIG-Bench Hard	95	26.0	36.0	41.0
BoolQ	96	4.0	4.7	3.9
CommonSenseQA	30	0.1	0.8	0.6
DROP	–	–	–	–
GSM8K	41	0.0	0.1	1.3
HellaSwag	85	14.8	14.8	14.3
HumanEval	–	–	–	–
MATH	1	0.0	-0.1	-0.2
MBPP	–	–	–	–
MMLU	–	–	–	–
MMLU-Pro	–	–	–	–
NaturalQuestions	52	1.6	0.9	0.8
OpenBookQA	21	3.0	3.3	2.6
PiQA	55	8.5	7.9	8.1
QuaC	99	2.4	11.0	6.4
RACE	–	–	–	–
SiQA	63	2.0	2.3	2.6
SQuAD	0	0.0	0.0	0.0
Winogrande	6	-0.1	-0.1	-0.2
WorldSense	73	-3.1	-0.4	3.9

- How to decide which examples are contaminated?
  - "An example of a dataset D to be contaminated if a ratio T_D of its tokens are part of an 8-gram occurring at least once in the pre-training corpus"
- How to decide estimated performance gains from contamination?
  - Compare the performance on the "clean" subset vs entire dataset

Other LLMs

Open/Closed Access

- **Weights:** open? described? closed?
- **Inference Code:** open? described? closed?
- **Training Code:** open? described? closed?
- **Data:** open? described? closed?

Licenses and Permissiveness

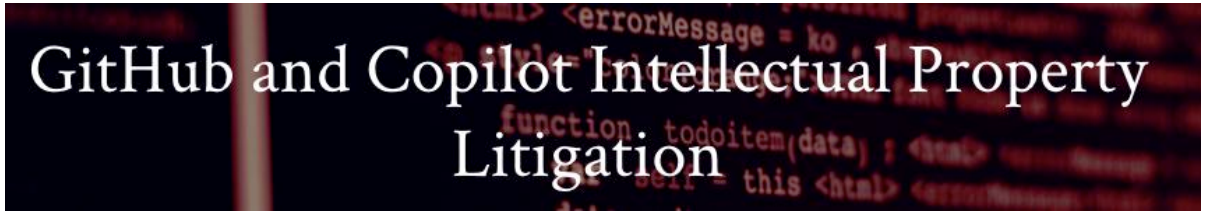
- **Public domain, CC-0:** old copyrighted works and products of US government workers
- **MIT, BSD:** very few restrictions
- **Apache, CC-BY:** must acknowledge owner
- **GPL, CC-BY-SA:** must acknowledge and use same license for derivative works
- **CC-NC:** cannot use for commercial purposes
- **LLaMa, OPEN-RAIL:** various other restrictions
- **No License:** all rights reserved, but can use under fair use

Fair Use

- US **fair use** doctrine — can use copyrighted material in some cases
- A gross simplification:
  - **Quoting** a small amount of material → likely OK
  - **Doesn't diminish** commercial value → possibly OK
  - Use for **non-commercial** purposes → possibly OK
- Most data on the internet is copyrighted, so model training is currently done assuming fair use
- But there are lawsuits!


The Times Sues OpenAI and Microsoft Over A.I. Use of Copyrighted Work

Millions of articles from The New York Times were used to train chatbots that now compete with it, the lawsuit said.



Why Restrict Model Access?

- **Commercial Concerns:** Want to make money from the models
- **Safety:** Limited release prevents possible misuse
- **Legal Liability:** Training models on copyrighted data is a legal/ethical gray area

English-Centric Open Models

- Open source/reproducible:
  - **Pythia:** Fully open, many sizes/checkpoints
  - **OLMo:** Possibly strongest reproducible model
- Open weights:
  - **LLaMa1/2/3/3.1:** Most popular, heavily safety tuned
  - **Mistral/Mixtral:** Strong and fast model, several European languages
  - **Qwen:** Strong, more multilingual - particularly en/zh

Pythia: Overview

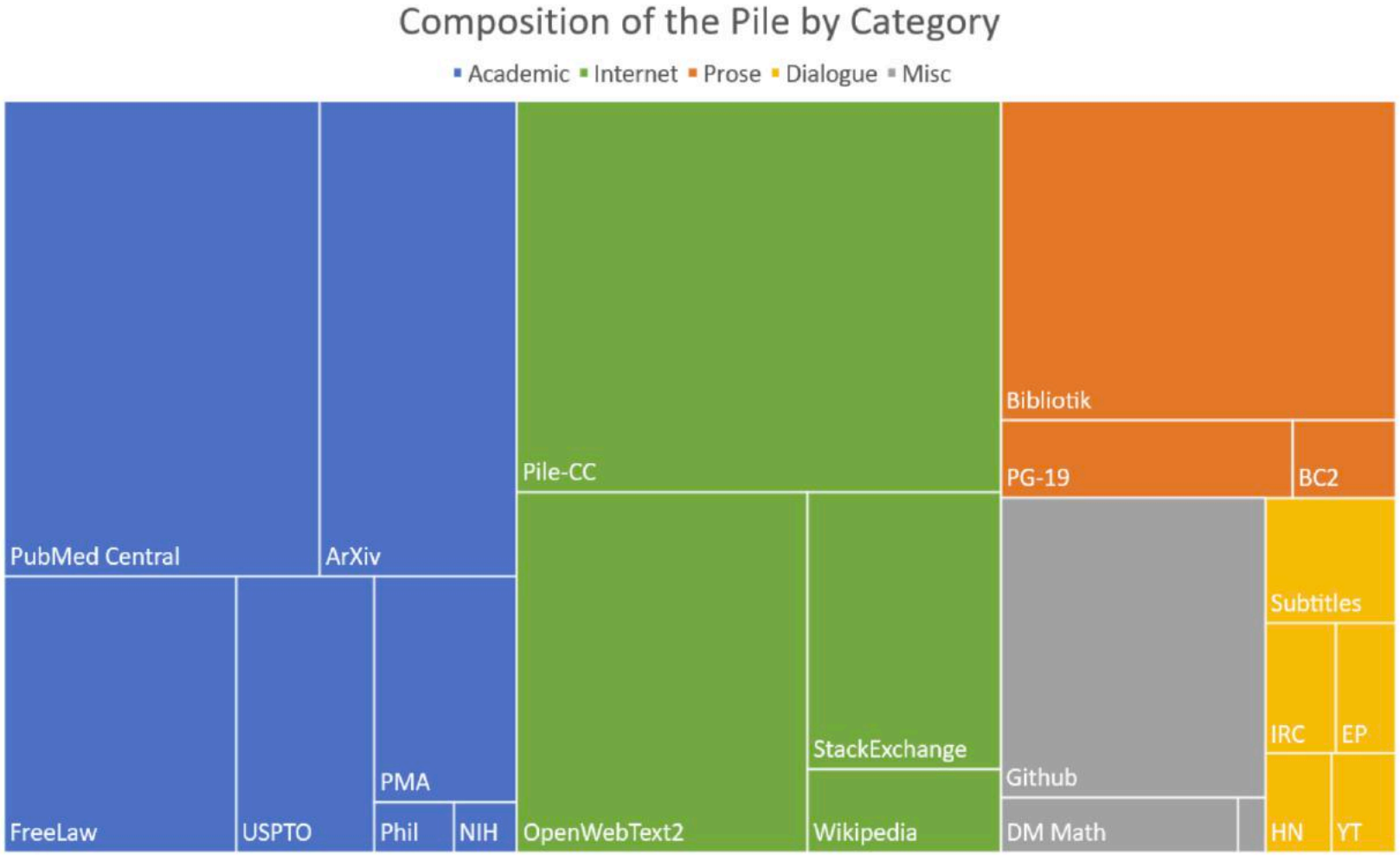
- **Creator:** 

- **Goal:** Joint understanding of model training dynamics and scaling
- **Unique features:** 8 model sizes 70M-12B, 154 checkpoints for each

Arch	Transformer+RoPE+SwiGLU, context 2k (cf LLaMa 4k), parametric LN
Data	Trained on 300B tokens of The Pile (next slide), or deduped 207B
Train	LR scaled inversely to model size (7B=1.2e-4), batch size 2M tokens

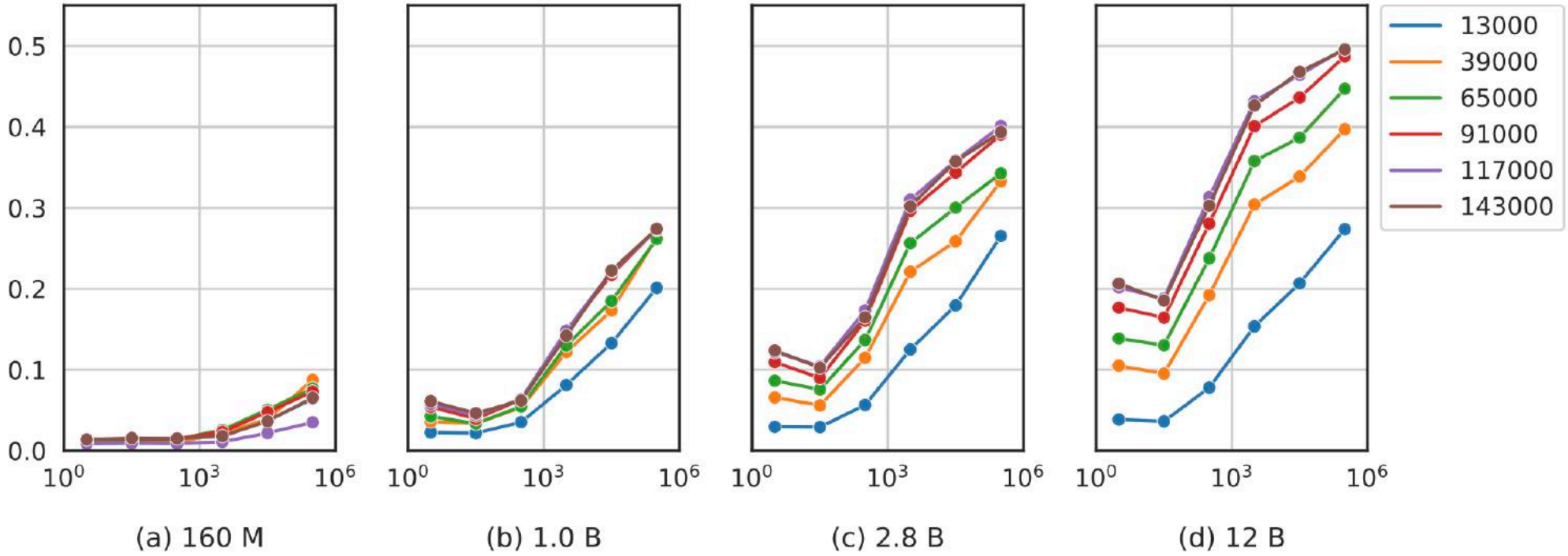
The Pile

- A now-standard 800GB dataset of lots of text/code



Pythia: Findings

- Some insights into training dynamics, e.g. larger models memorize facts more quickly (x axis: fact frequency, legend: training step)



- It is possible to intervene on data to reduce gender bias

OLMo: Overview

- **Creator:** AI2 Allen Institute for AI
- **Goal:** Better science of state-of-the-art LMs
- **Unique features:** Top performance of fully documented model, instruction tuned etc.

Arch	Transformer+RoPE+SwiGLU, context 4k, non-parametric LN
Data	Trained on 2.46T tokens of Dolma corpus (next slide)
Train	LR scaled inversely to model size (7B=3e-4), batch size 4M tokens

Dolma

- 3T token corpus created and released by AI2 for LM training
- A pipeline of (1) language filtering, (2) quality filtering, (3) content filtering, (4) deduplication, (5) multi-source mixing, and (6) tokenization

Source	Doc Type	UTF-8 bytes *(GB)*	Documents *(millions)*	Unicode words *(billions)*	Llama tokens *(billions)*
Common Crawl	web pages	9,022	3,370	1,775	2,281
The Stack	code	1,043	210	260	411
C4	web pages	790	364	153	198
Reddit	social media	339	377	72	89
PeS2o	STEM papers	268	38.8	50	70
Project Gutenberg	books	20.4	0.056	4.0	6.0
Wikipedia, Wikibooks	encyclopedic	16.2	6.2	3.7	4.3
Total		**11,519**	**4,367**	**2,318**	**3,059**

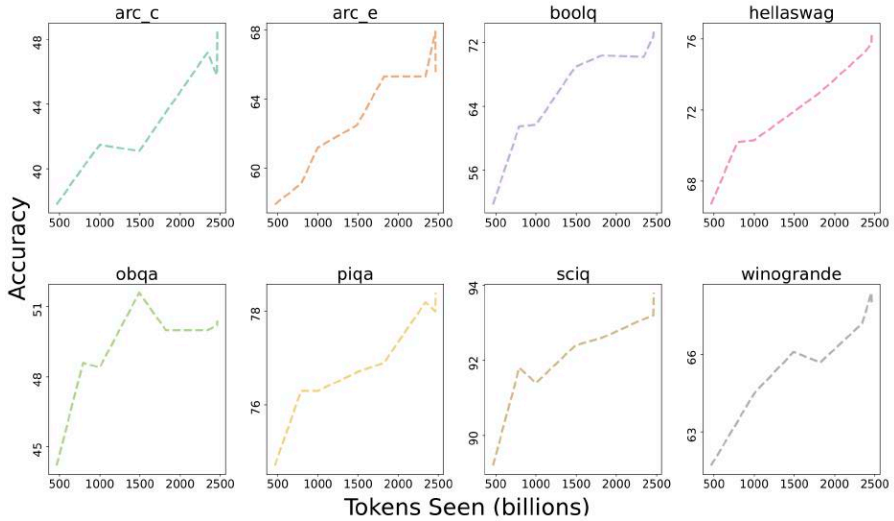
OLMo: Findings

- Competitive average performance

7B Models	arc challenge	arc easy	boolq	hella-swag	open bookqa	piqa	sciq	wino-grande	avg.
Falcon	47.5	70.4	74.6	75.9	53.0	78.5	93.9	68.9	70.3
LLaMA	44.5	67.9	75.4	76.2	51.2	77.2	93.9	70.5	69.6
Llama 2	48.5	69.5	80.2	76.8	48.4	76.7	94.5	69.4	70.5
MPT	46.5	70.5	74.2	77.6	48.6	77.3	93.7	69.9	69.8
Pythia	44.1	61.9	61.1	63.8	45.0	75.1	91.1	62.0	63.0
RPJ-INCITE	42.8	68.4	68.6	70.3	49.4	76.0	92.9	64.7	66.6
OLMo-7B	48.5	65.4	73.4	76.4	50.4	78.4	93.8	67.9	69.3

Table 6: Zero-shot evaluation of OLMo-7B and 6 other publicly available comparable model checkpoints on 8 core tasks from the downstream evaluation suite described in Section 2.4. For OLMo-7B, we report results for the 2.46T token checkpoint.

- Performance increases constantly w/ training



Llama2: Overview

- **Creator:** Meta
- **Goal:** Strong and safe open LM w/ base+chat versions
- **Unique features:** Open model with strong safeguards and chat tuning, good performance

Arch	Transformer+RoPE+SwiGLU, context 4k, RMSNorm
Data	Trained on “public sources, up-sampling the most factual sources”, LLaMa 1 has more info (next page), total 2T tokens
Train	7B=3e-4, batch size 4M tokens

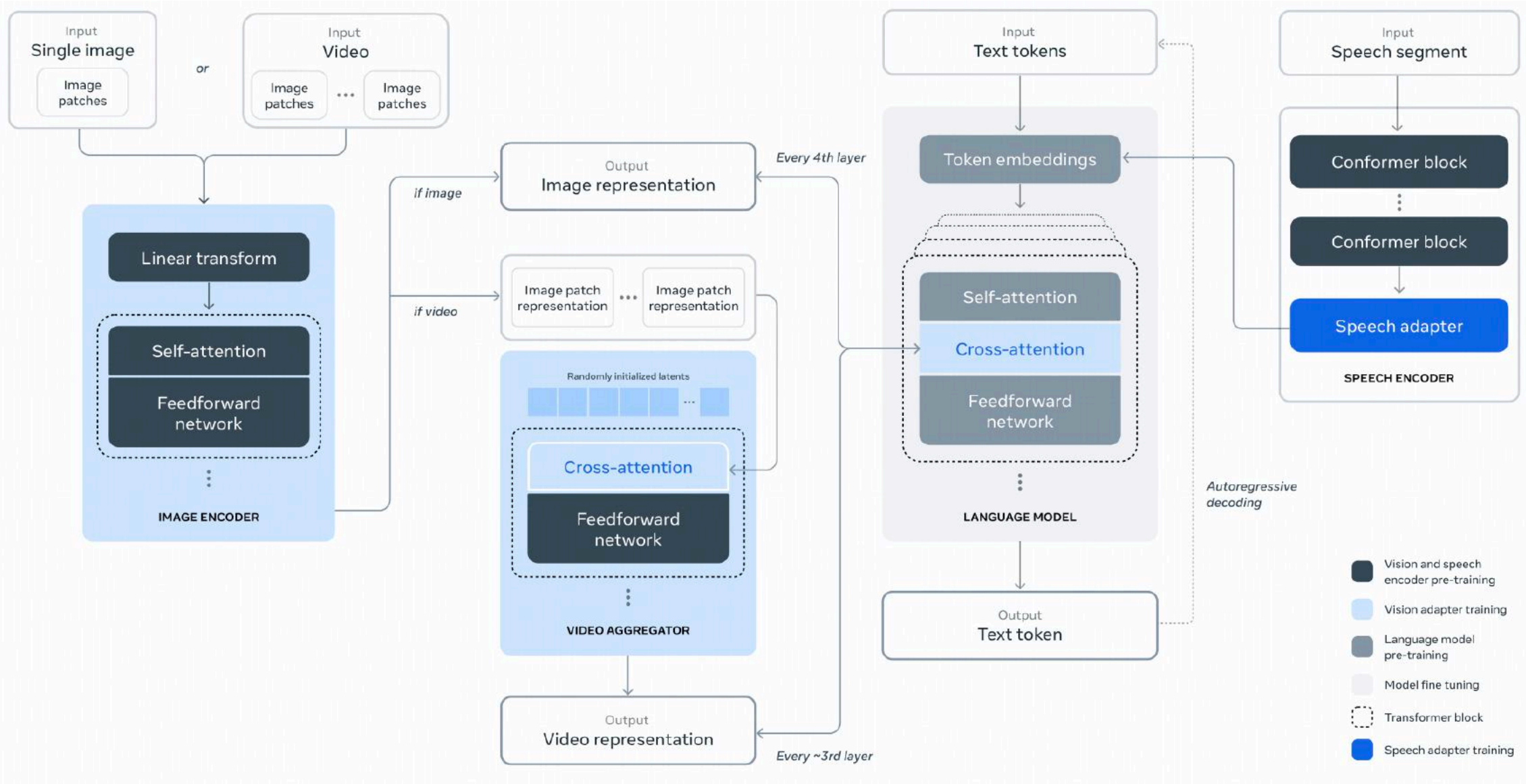
Llama3.1: Overview

- **Creator:** Meta
- **Goal:** A herd of language models that natively support multilinguality, coding, reasoning, and tool usage
- **Compared with Llama2:** Larger Data scale (15T multilingual tokens vs 1.8T tokens). More Training FLOPs (3.8 × 10^25 FLOPs, almost 50× more than the largest version of Llama 2)

GPUs	TP	CP	PP	DP	Seq. Len.	Batch size/DP	Tokens/Batch	TFLOPs/GPU	BF16 MFU
8,192	8	1	16	64	8,192	32	16M	430	43%
16,384	8	1	16	128	8,192	16	16M	400	41%
16,384	8	16	16	4	131,072	16	16M	380	38%

Table 4 Scaling configurations and MFU for each stage of Llama 3 405B pre-training. See text and Figure 5 for descriptions of each type of parallelism.

Llama3.1: Multimodality



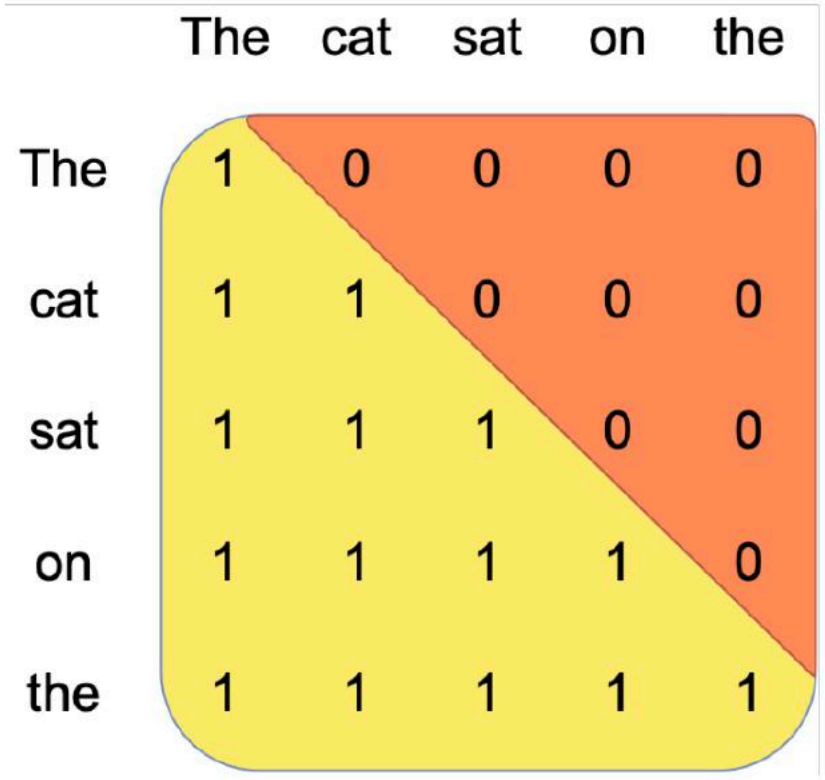
Mistral/Mixtral: Overview

- **Creator:** 

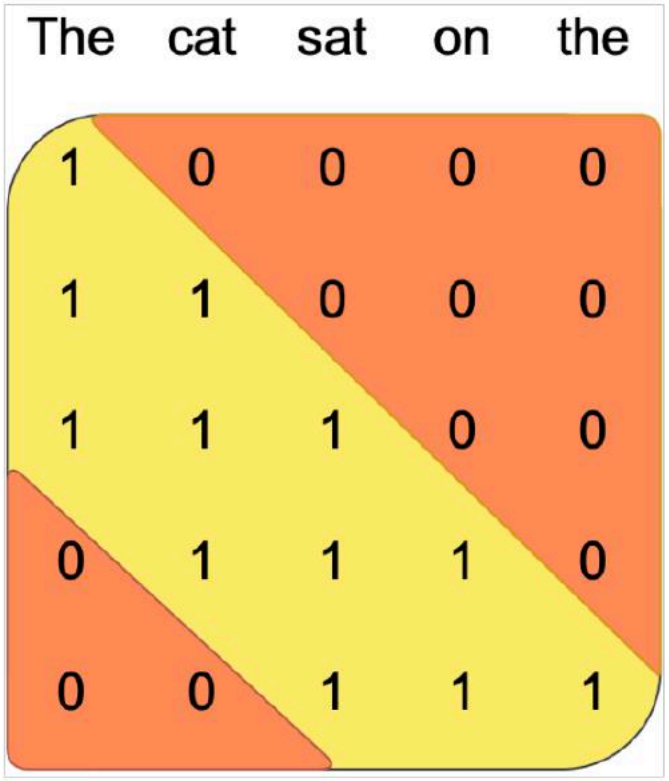
- **Goal:** Strong and somewhat multilingual open LM
- **Unique features:** Speed optimizations, including GQA and Mixture of Experts

Arch	Transformer+RoPE+SwiGLU, context 4k, RMSNorm, sliding window attention. Mixtral has 8x experts in feed-forward layer
Data	Not disclosed? But includes English and European languages
Train	Not disclosed?

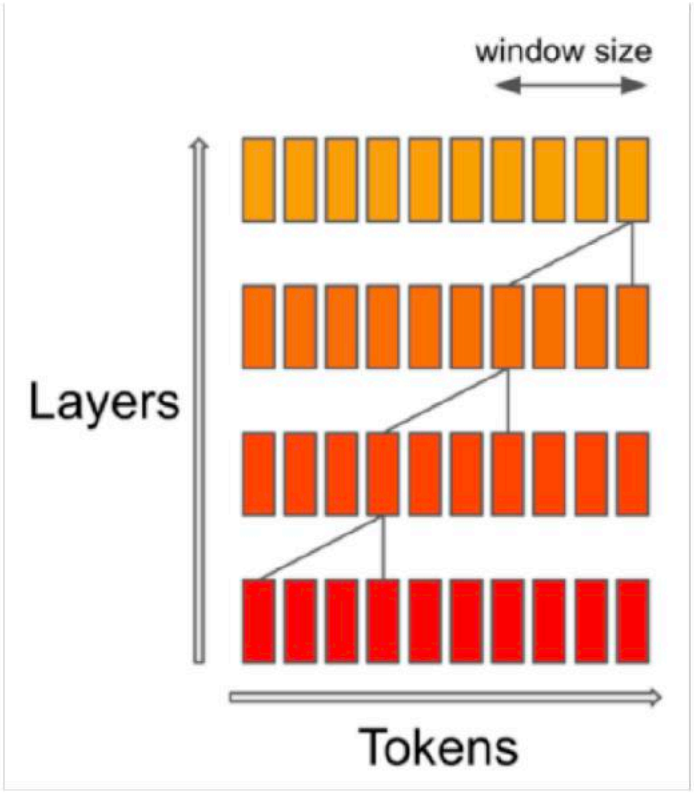
Mistral: Sliding Window Attention



Vanilla Attention



Sliding Window Attention



Effective Context Length

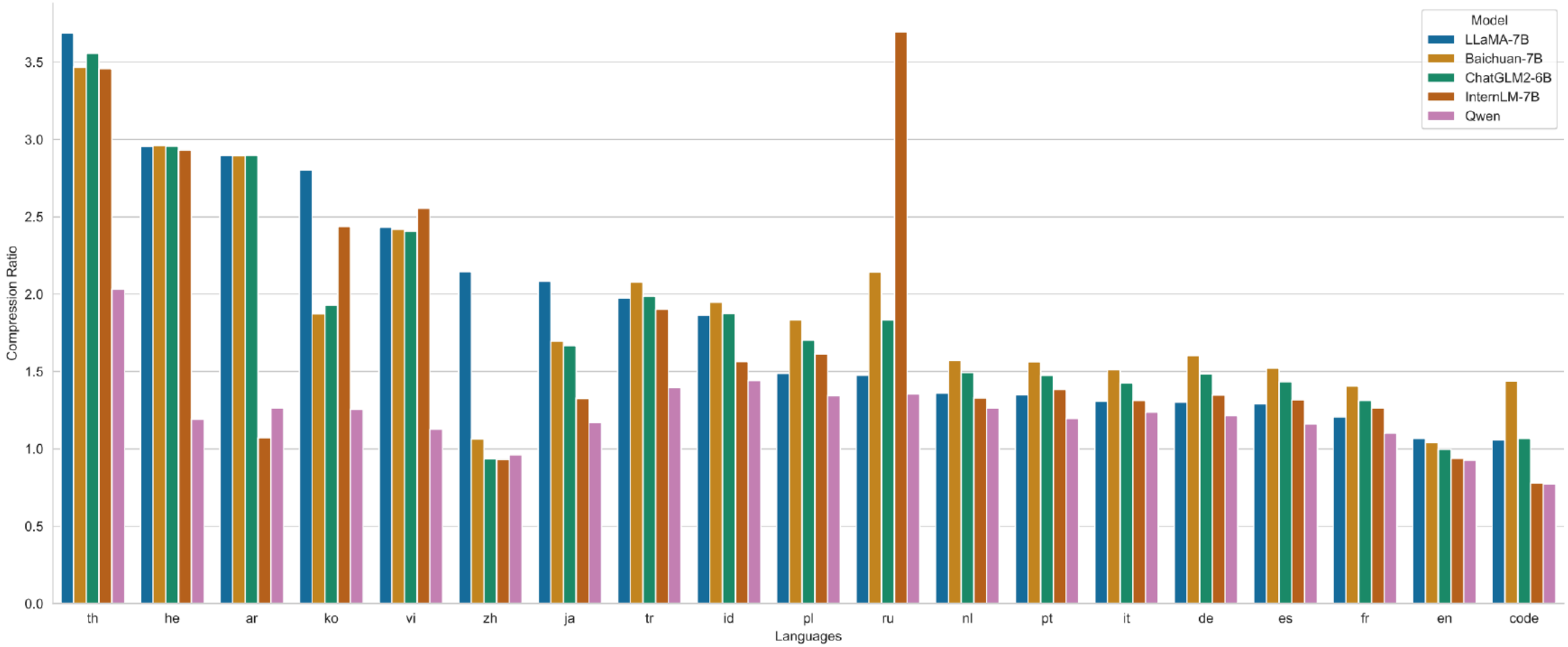
Qwen: Overview

- **Creator:** Alibaba
- **Goal:** Strong multilingual (esp. English and Chinese) LM
- **Unique features:** Large vocabulary for multilingual support, strong performance

Arch	Transformer+RoPE+SwiGLU, context 4k, RMSNorm, bias in attention layer
Data	Trained on multilingual data + instruction data at pre-training time, 2-3T tokens
Train	3e-4, batch size 4M tokens

Qwen: Multilinguality

- Token compression ratio re: XLM-R (lower is better)



SmolLM: Overview

- **Creator:**  **Hugging Face**
- **Goal:** Small scale (135M, 360M, and 1.7B parameters) but strong performance
- **Unique features:** Fully Open-sourced with a high-quality pre-training corpus.

- **Cosmopedia v2**: A collection of synthetic textbooks and stories generated by Mixtral (28B tokens)
- **Python-Edu**: educational Python samples from The Stack (4B tokens)
- **FineWeb-Edu (deduplicated)**: educational web samples from FineWeb (220B tokens)

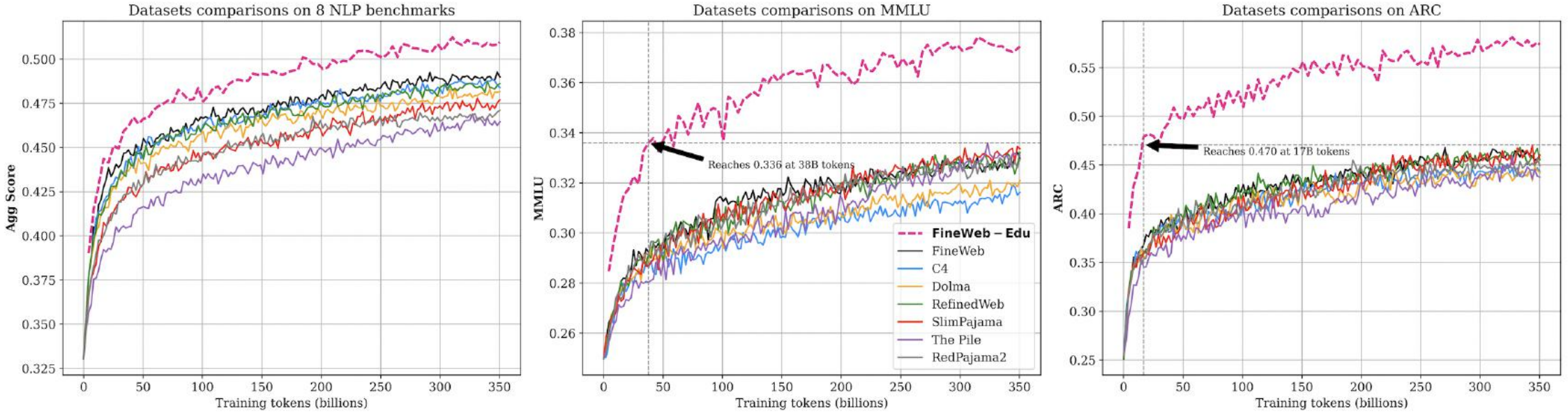
https://huggingface.co/blog/smollm

FineWeb – (Edu)

🍷 FineWeb dataset consists of more than 15T tokens of cleaned and deduplicated english web data from CommonCrawl.

Url Filtering → Trafilatura text extraction from HTML → FastText LanguageFilter → Quality filtering → MinHash deduplication → PII Formatting

“To enhance FineWeb's quality, we developed an educational quality classifier using annotations generated by LLama3-70B-Instruct. We then used this classifier to retain only the most educational web pages.”



Other Models

Code Models

- **StarCoder 2** — by Big Science (leads: Hugging Face + Service Now), fully open model
- **CodeLlama** — by Meta, code adaptation of LLaMa
- **DeepSeek Coder** — by DeepSeek, strong performance across many tasks
- **Yi Coder** - by 01.AI, smaller scales (9B/1.5B) but strong performance.
- More in code generation class!

Math Models

- **LLEMA** — by EleutherAI and others, model for math theorem proving trained on proof pile
- **DeepSeek Math** — by DeepSeek, finds math-related pages on the web

Science Model: Galactica

- Model for science trained by Meta
- Diverse set of interesting training data

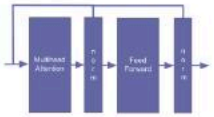
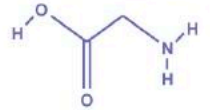
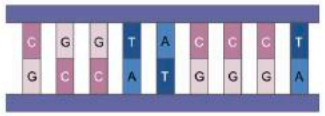
Modality	Entity	Sequence	
Text	Abell 370	`Abell 370 is a cluster...`	
LaTeX	Schwarzschild radius	`r_{s} = \frac{2GM}{c^2}`	$r_s = \frac{2GM}{c^2}$
Code	Transformer	`class Transformer(nn.Module)`	
SMILES	Glycine	`C(C(=O)O)N`	
AA Sequence	Collagen α-1(II) chain	`MIRLGAPQTL..`	
DNA Sequence	Human genome	`CGGTACCCTC..`	

Table 1: Tokenizing Nature. Galactica trains on text sequences that represent scientific phenomena.

Closed Models

GPT-4o: Overview

- **Creator:** OpenAI
- De-facto standard “strong” language model
- Tuned to be good as a chat-based assistant
- Supports calling external tools through “function calling” interface
- Accepts image inputs
- Fast and cheaper inference compared with earlier GPT-4 versions

Gemini

- **Creator:** 

- Performance competitive with corresponding GPT models (Gemini Pro 1.0 ~ gpt-3.5, Gemini Ultra 1.0 ~ gpt-4)
- Pro 1.5 supports very long inputs, 1-10M tokens
- Supports image and video inputs
- Can generate images natively

Claude 3: Overview

- **Creator:** ANTHROP\C
- Context window up to 200k
- Allows for processing images
- Overall strong results competitive with GPT-4

Long Context Models

How Long are Sequences?

- One sentence: ~20 tokens
- One document: 100-10k tokens
- One book: 50k-300k tokens
- One video: 1.5k-1M tokens (~300/sec)
- One codebase: 20k-1B tokens
- One genome: 3B nucleotides

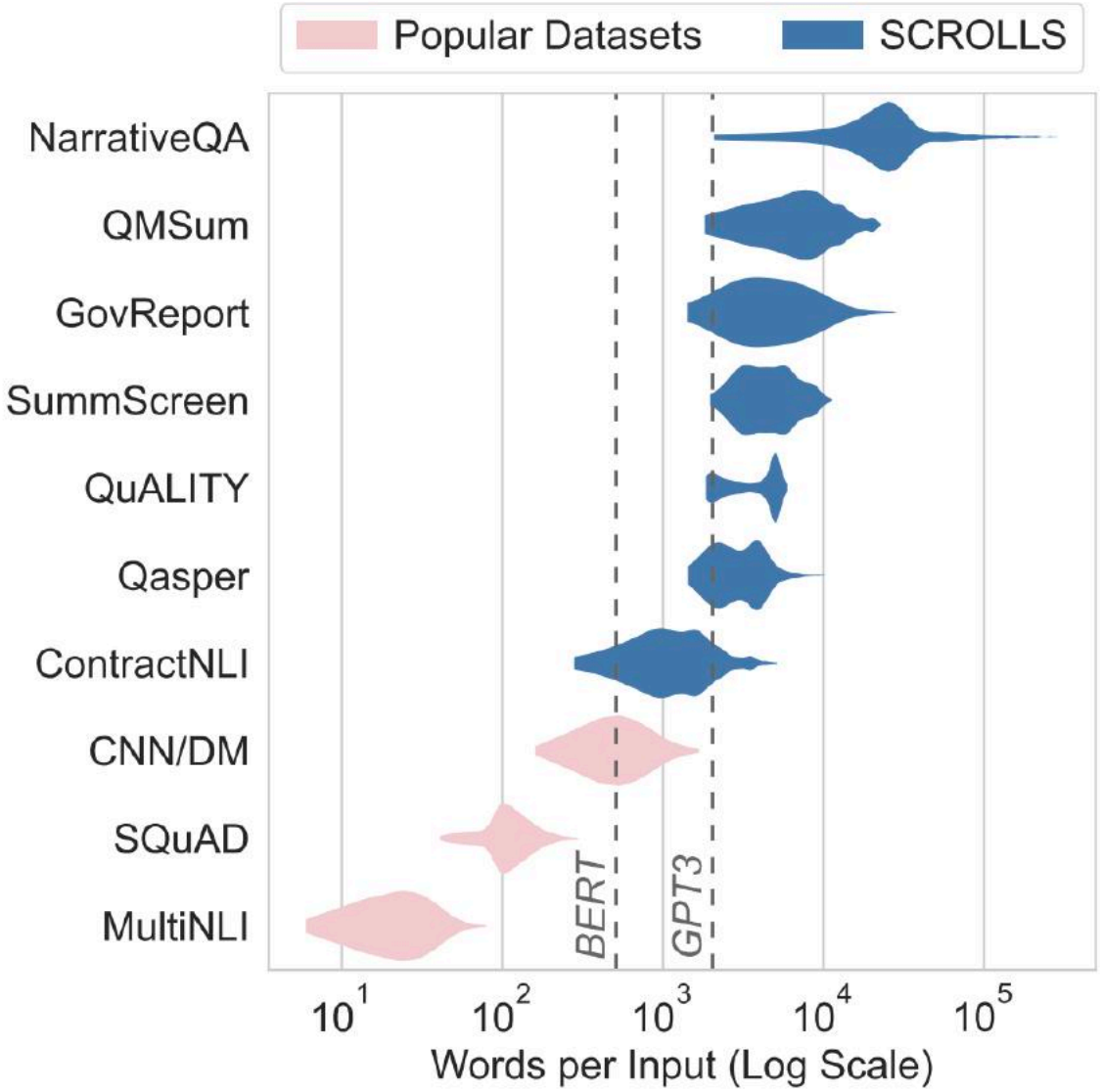
Why is Modeling Long Sequences Hard?

- **Memory Complexity:** Transformer models scale quadratically in memory
- **Compute Complexity:** Transformer models scale quadratically in computation
- **Training:** Data is lacking, training signal is weak, training on long sequences is costly

Long-Context Use Cases and Evaluation

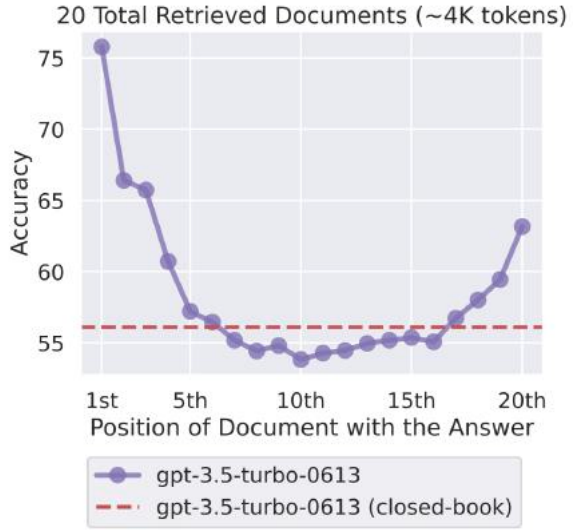
Benchmarks for Long-context Models

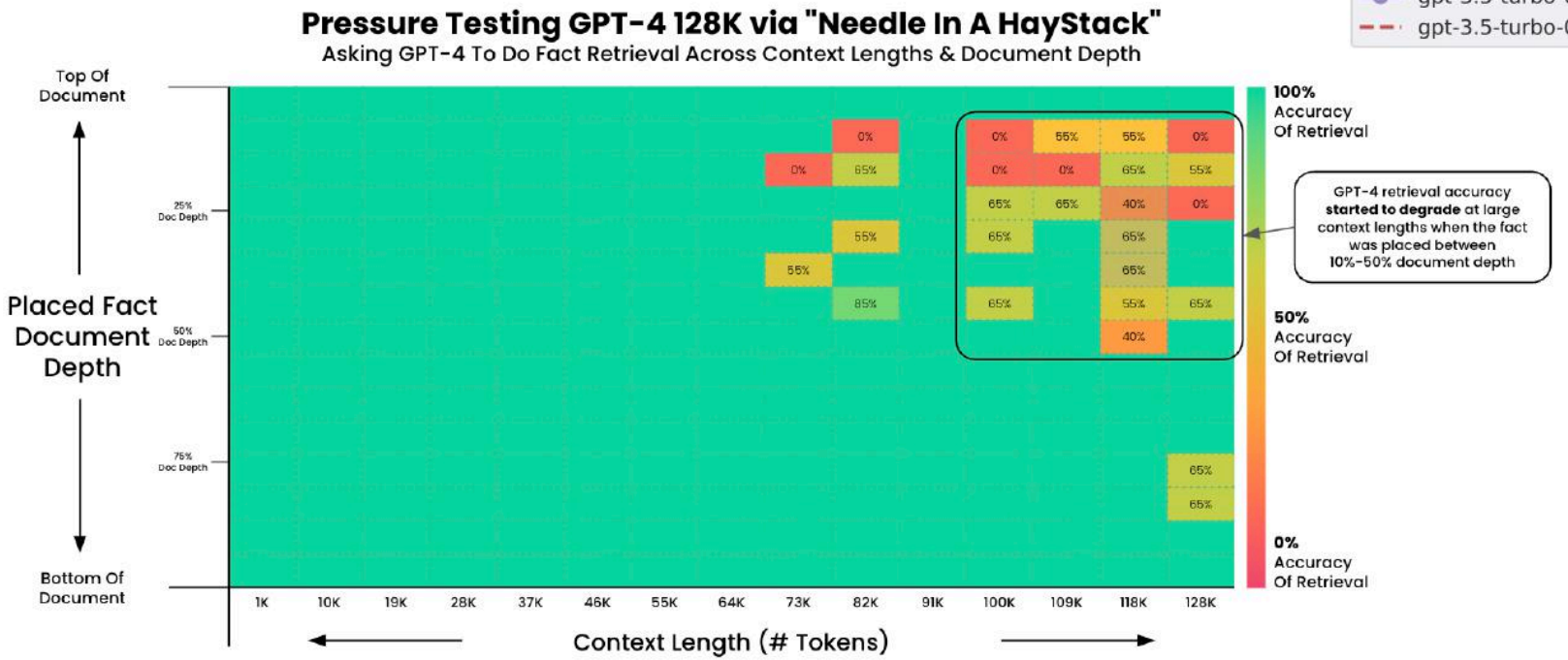
- **Long Range Arena:**
  Composite benchmark containing mostly non-NLP tasks (Tay et al. 2020)
- **SCROLLS:**
  Benchmark containing long-context summarization, QA, etc. (Shaham et al. 2022)



Targeted Analysis Tools

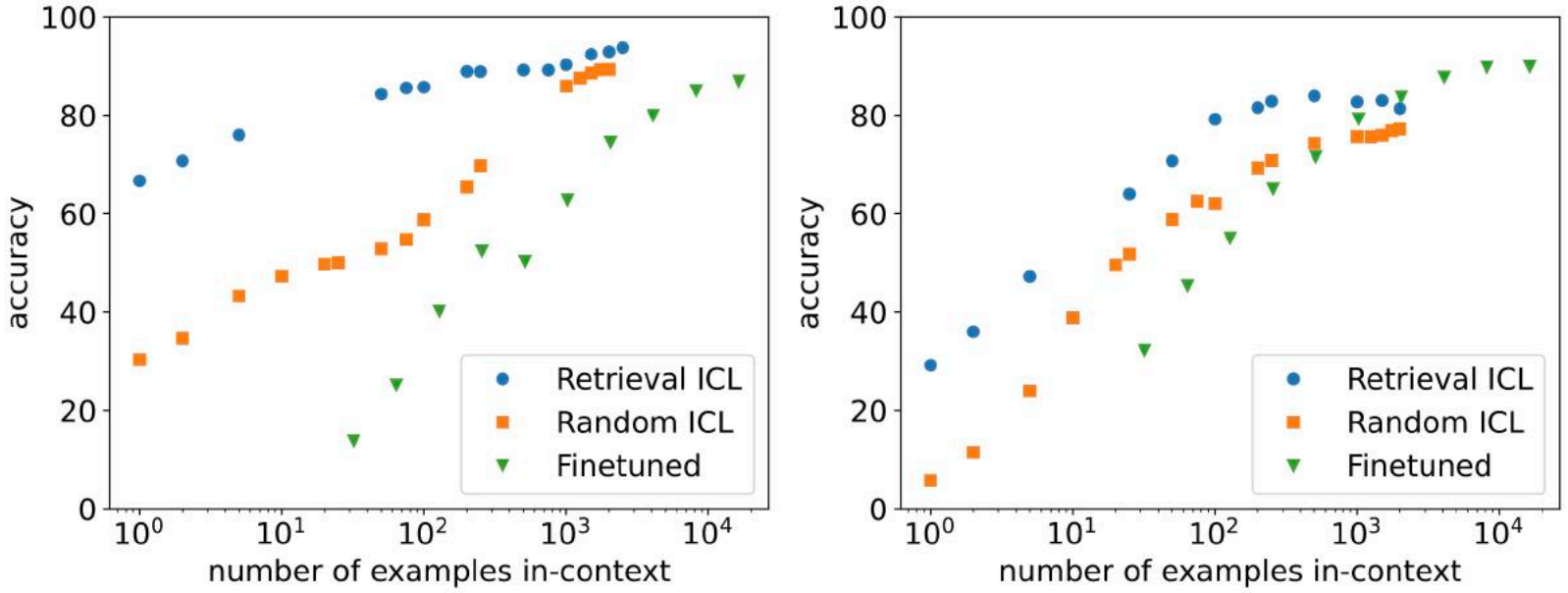
- "lost-in-the-middle" (Liu et al. 2023) demonstrates that models pay less attention to things in middle context
- "needle in a haystack" tests (Kamradt 2023) test across document length/position
- RULER (Hsieh et al. 2024) compiles a number of different NIAH tasks





Long-context In-context Learning (Bertsch et al. 2024)

- Can we provide lots of examples to long-context models and improve accuracy through ICL?



(a) Clinic-150

(b) Trecfine

- When many in-context examples are provided, it can be better than fine-tuning!

Long-context Dialog

- Chatbots that maintain long-term conversational context
- e.g. Locomo corpus (Maharana et al. 2024)
- Evaluate w/ question answering, summarization, response generation

Tackling Complexity: Memory-efficient Computation

Vanilla Attention Complexity

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

$$A = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) \qquad \text{Attention}(Q, K, V) = AV$$

Time: $O(bs^2d)$ for QK^T
(but fast on GPU)
Memory: $O(bs^2)$ for all ops

Time: $O(bs^2d)$ for AV
(but fast on GPU)
Memory: $O(bsd)$

b: batch size, s: sequence length, d: dimension

Multi-head Attention Complexity

- Multi-head attention splits attention heads
- No effect on time complexity, but effect on memory

Time: $O(bs^2d)$ for QK^T
(but fast on GPU)
Memory: $O(bs^2h)$ for all ops

Time: $O(bs^2d)$ for AV
(but fast on GPU)
Memory: $O(bsd)$

b: batch size, s: sequence length, d: dimension

Memory-efficient Computation (Jang 2019, Rabe and Staats 2021)

- Insight: you don't need to materialize s^2 attention
- Calculate softmax numerator times values, and softmax denominator left-to-right

*softmax numerator * V*

$$V^* = \exp\left(\frac{QK^T}{\sqrt{d_k}}\right) V$$

Memory: O(bsd)

*softmax denominator * V*

$$S^* = \text{sum}\left(\exp\left(\frac{QK^T}{\sqrt{d_k}}\right)\right)$$

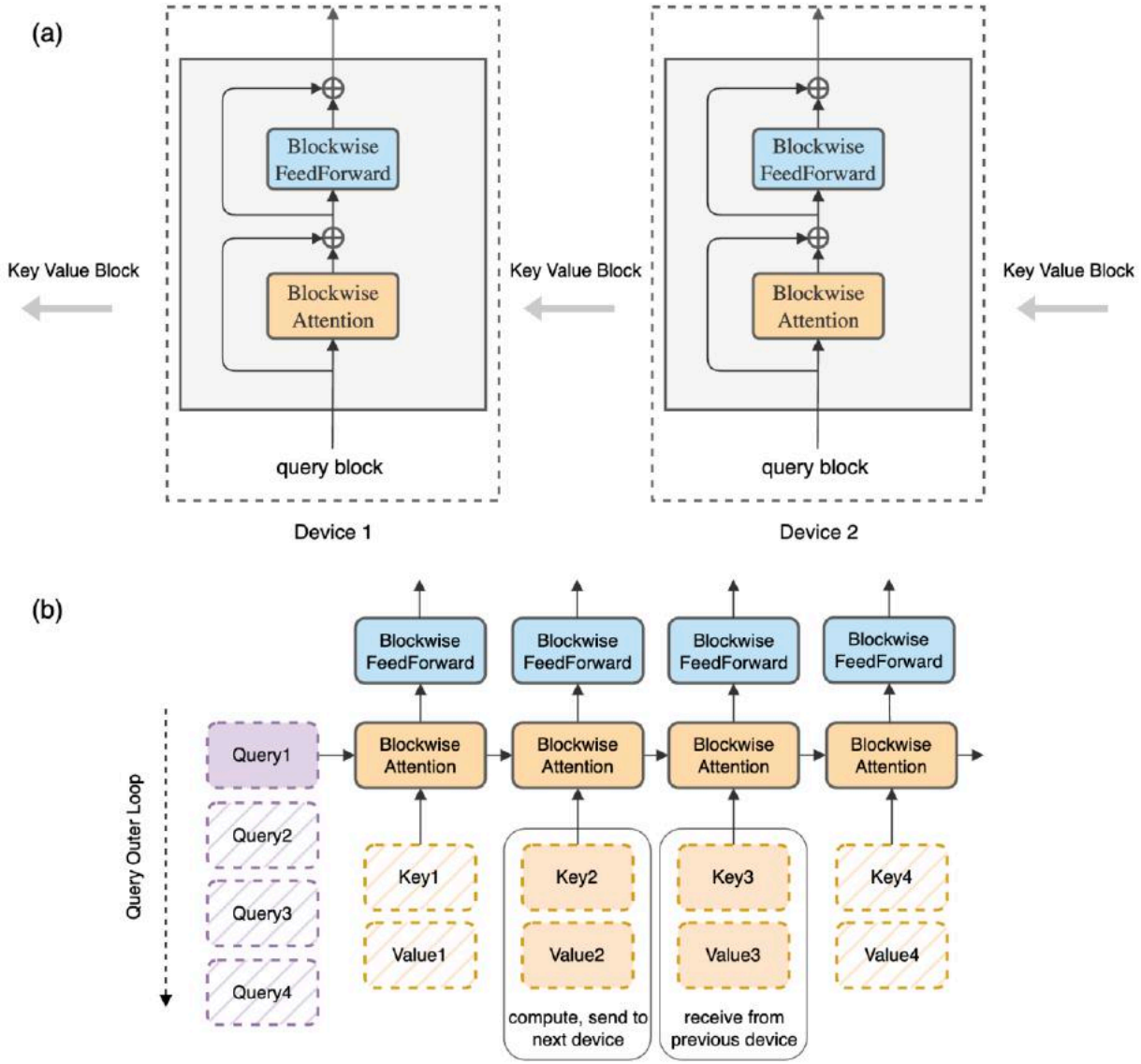
Memory: O(bsh)

$$\text{Attention}(Q, K, V) = V^*/S^*$$

Memory: O(bsd)

Ring Attention (Liu et al. 2023)

- Further distribute storage/incremental computation across multiple devices



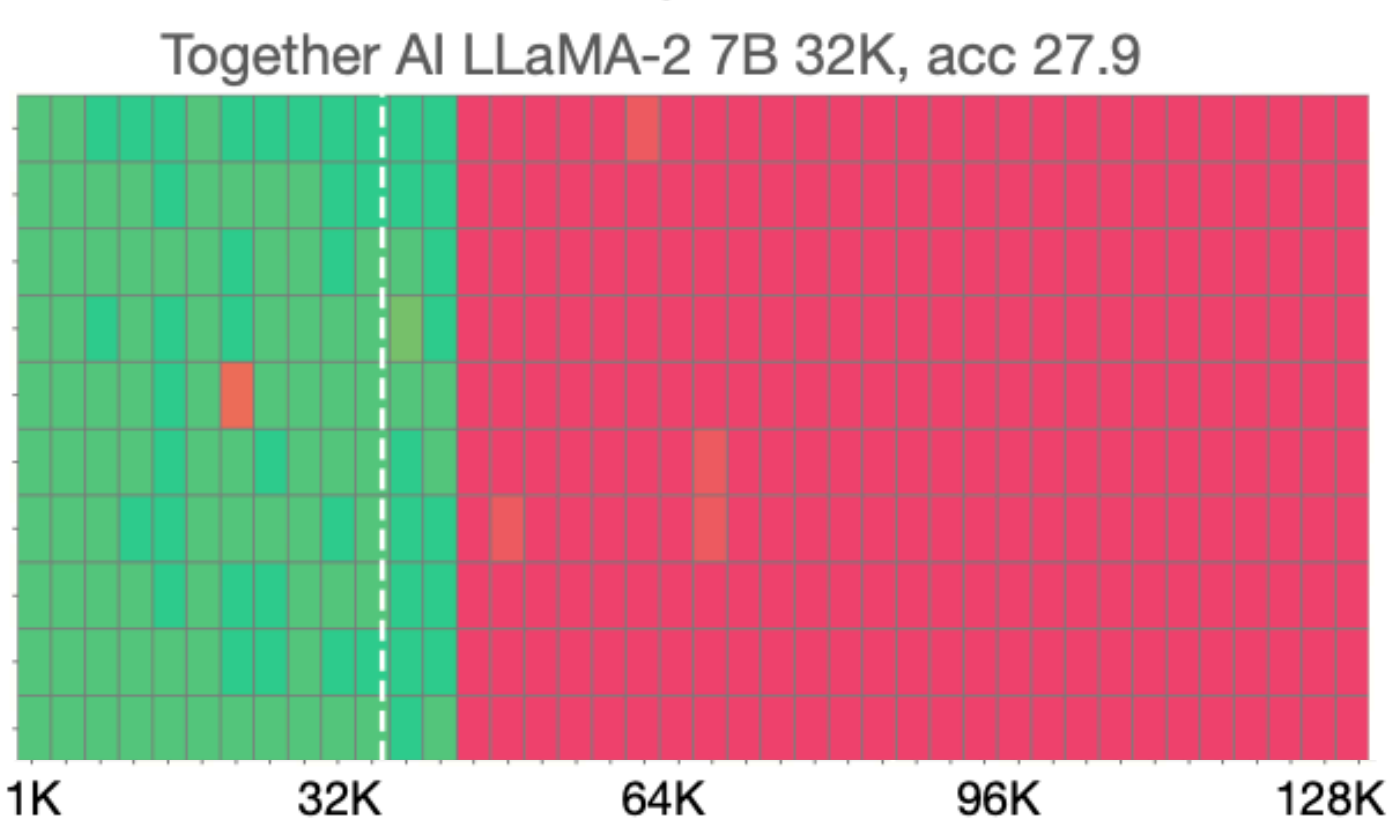
Extrapolation of Short-Context Models

Trained Models Fail to Extrapolate

- Most transformer models are trained on shorter sequences (4k)
  - If a document is longer than the limit, truncate or chunk
- This poses problems for positional encodings:
  - Learned absolute encodings: impossible to extrapolate
  - Fixed absolute encodings: move models out of distribution, very bad
  - Relative encodings: should extrapolate better in theory, but not really in practice

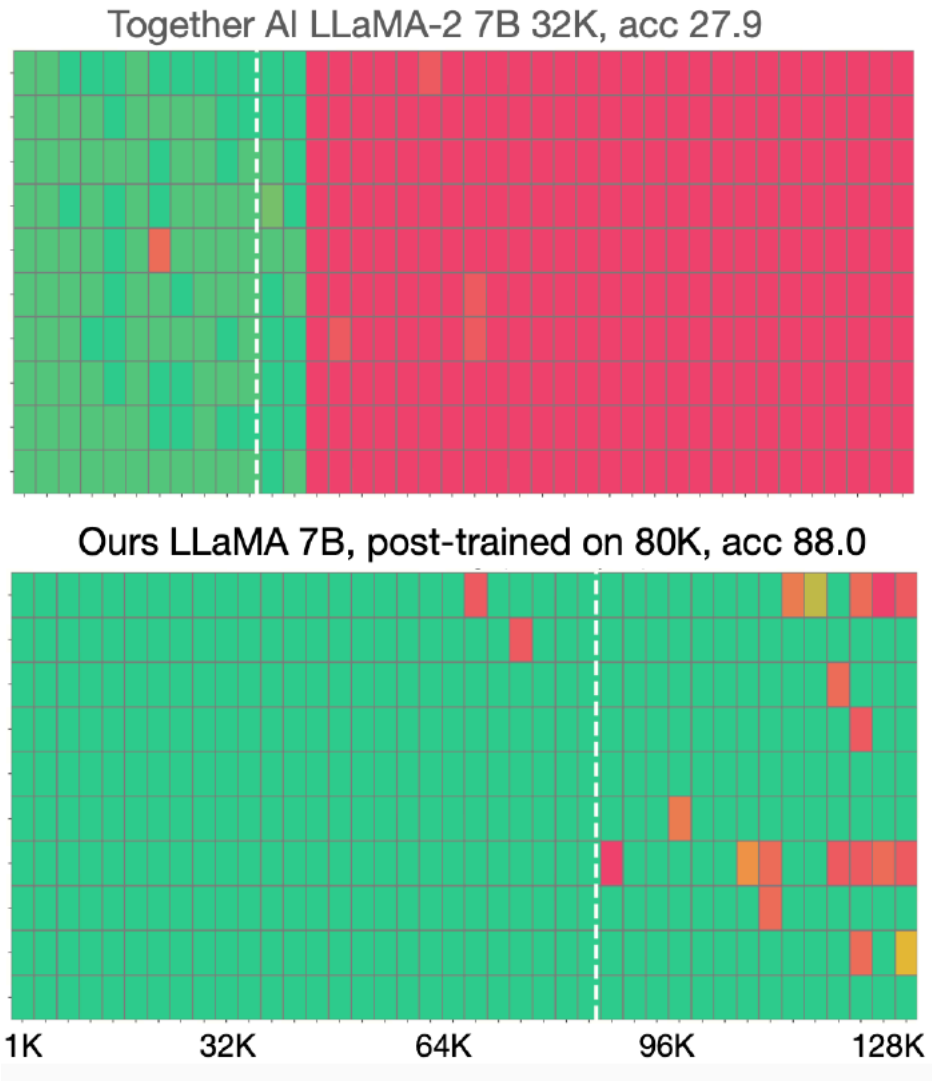
An Example of Failed Extrapolation (Fu et al. 2024)

- Llama-2 w/ 32k context (RoPE) can answer questions about sequences up to about 40k, but not beyond



Training w/ Long Context (Fu et al. 2024)

- Simple solution: continually train on longer documents
- **Problem:** there aren't many long documents
  - **Solution:** upsample the longer documents
- **Problem:** upsampling favors certain domains such as books and GitHub
  - **Solution:** maintain domain mixture, but upsample long docs in each domain



RoPE Scaling (Lu et al. 2024)

- RoPE has a parameter adjusting the period

$$\mathbf{R}(\boldsymbol{\theta}, i) = \begin{pmatrix} \cos i\theta_1 & -\sin i\theta_1 & \cdots & 0 & 0 \\ \sin i\theta_1 & \cos i\theta_1 & \cdots & 0 & 0 \\ \vdots & & & & \\ 0 & 0 & \cdots & \cos i\theta_{\frac{d_k}{2}} & -\sin i\theta_{\frac{d_k}{2}} \\ 0 & 0 & \cdots & \sin i\theta_{\frac{d_k}{2}} & \cos i\theta_{\frac{d_k}{2}} \end{pmatrix}$$

- typically $\theta_j = b^{-\frac{2j}{d_k}}$ with b=10000

- **Position interpolation:** Multiply θ by a constant scaling factor (e.g. C_{short}/C_{long})
- **Neural tangent kernel:** Scale low-frequency components, but maintain high-frequency components

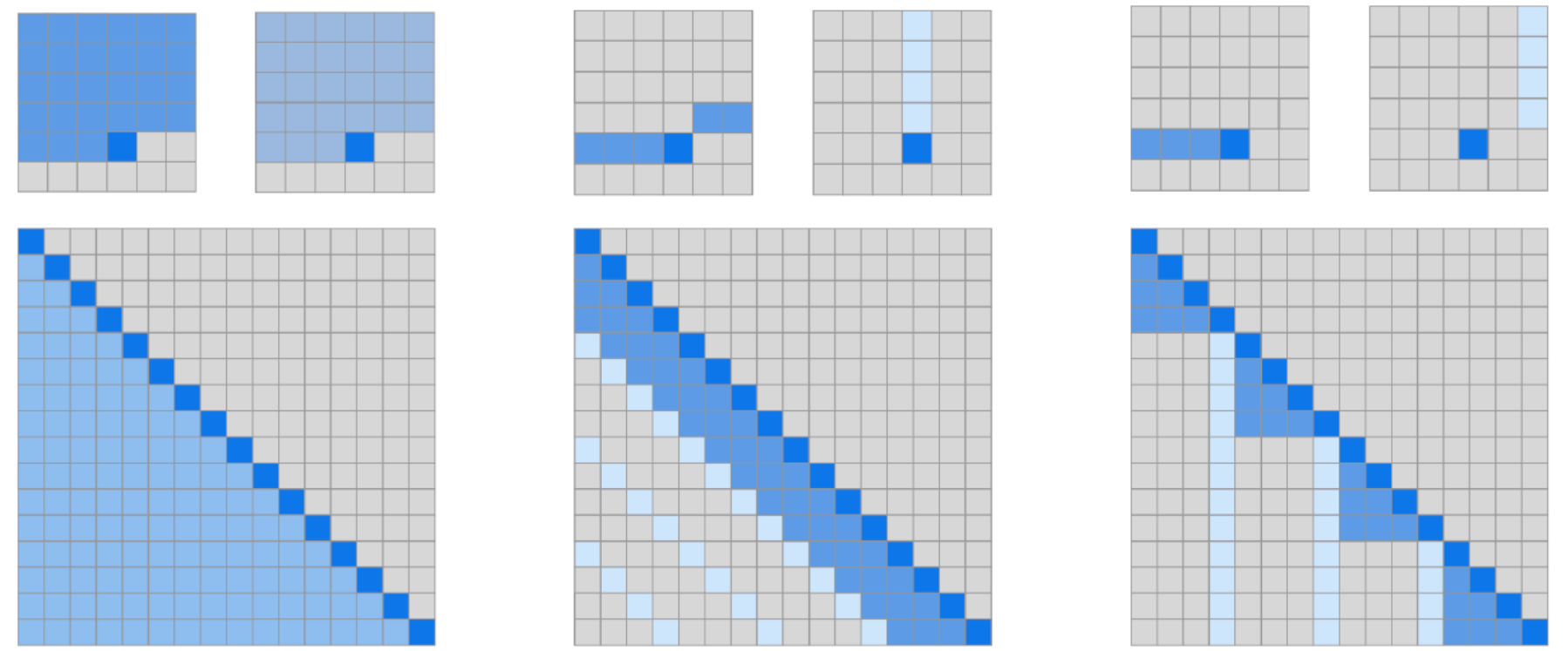
Tackling Complexity: Alternative Transformer Architectures

Tackling Transformer Complexity

- Sparse Attention
- Sliding Window Attention
- Compression
- Low-rank Approximation

Sparse Transformers (Child et al. 2019)

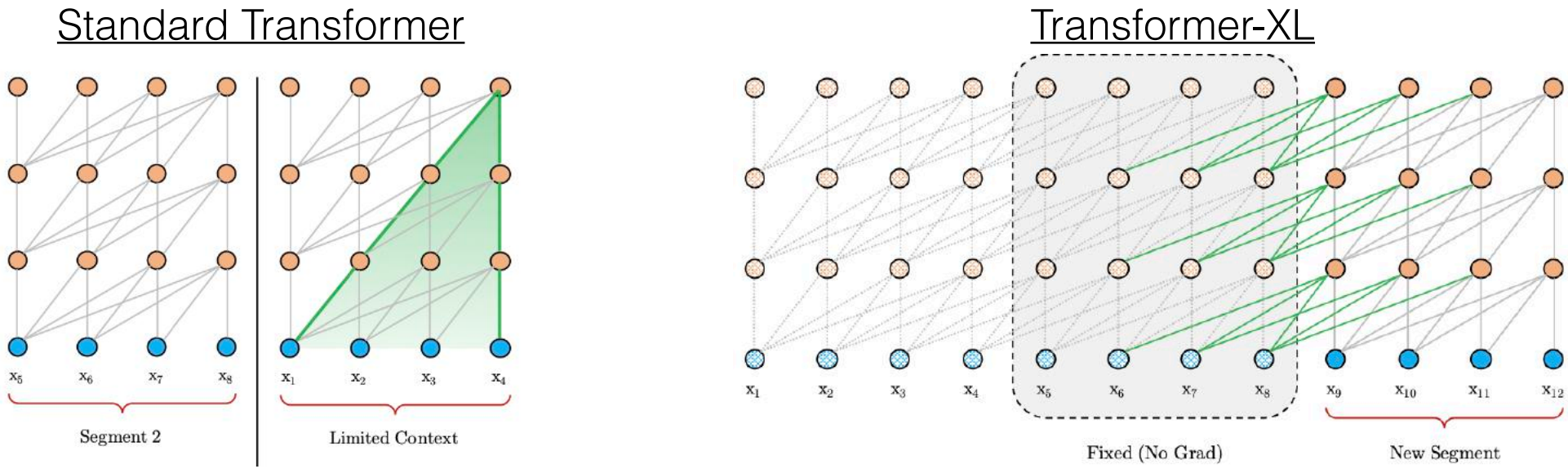
- Add "stride", only attending to every n previous states



(a) Transformer (b) Sparse Transformer (strided) (c) Sparse Transformer (fixed)

Truncated BPTT+Transformer

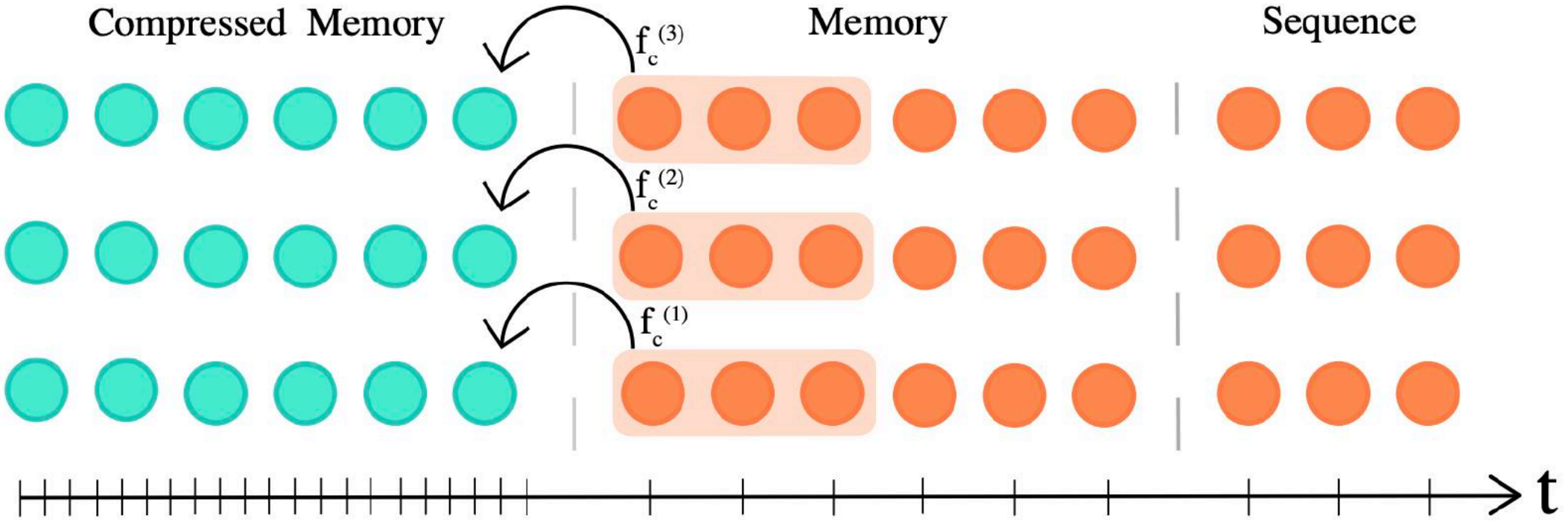
- Transformer-XL (Dai et al. 2019) attends to fixed **vectors** from the previous sentence



- Like truncated backprop through time for RNNs; can use previous states, but not backprop into them
- See also Mistral's (Jiang et al. 2023) sliding window attention

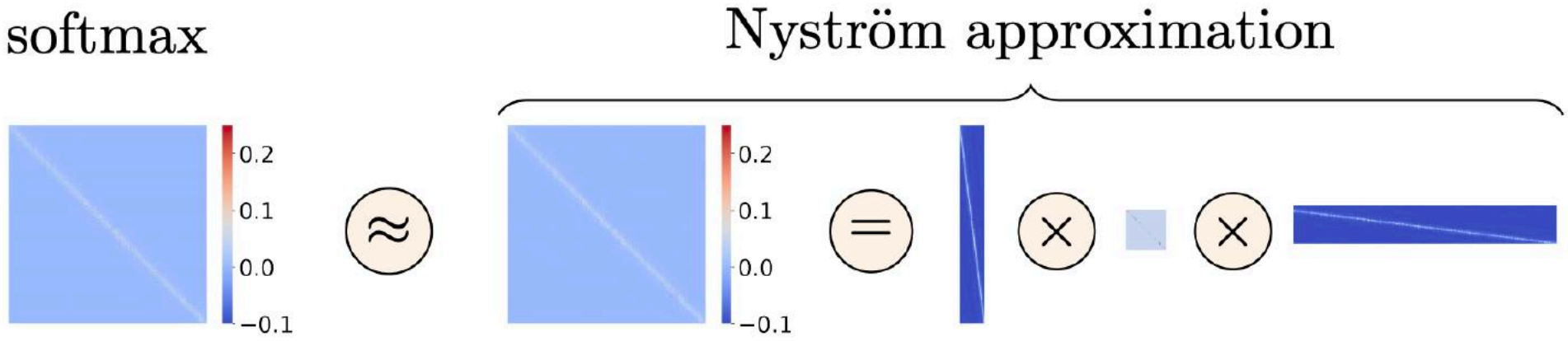
Compressing Previous States

- Add a "strided" compression step over previous states (Rae et al. 2019)



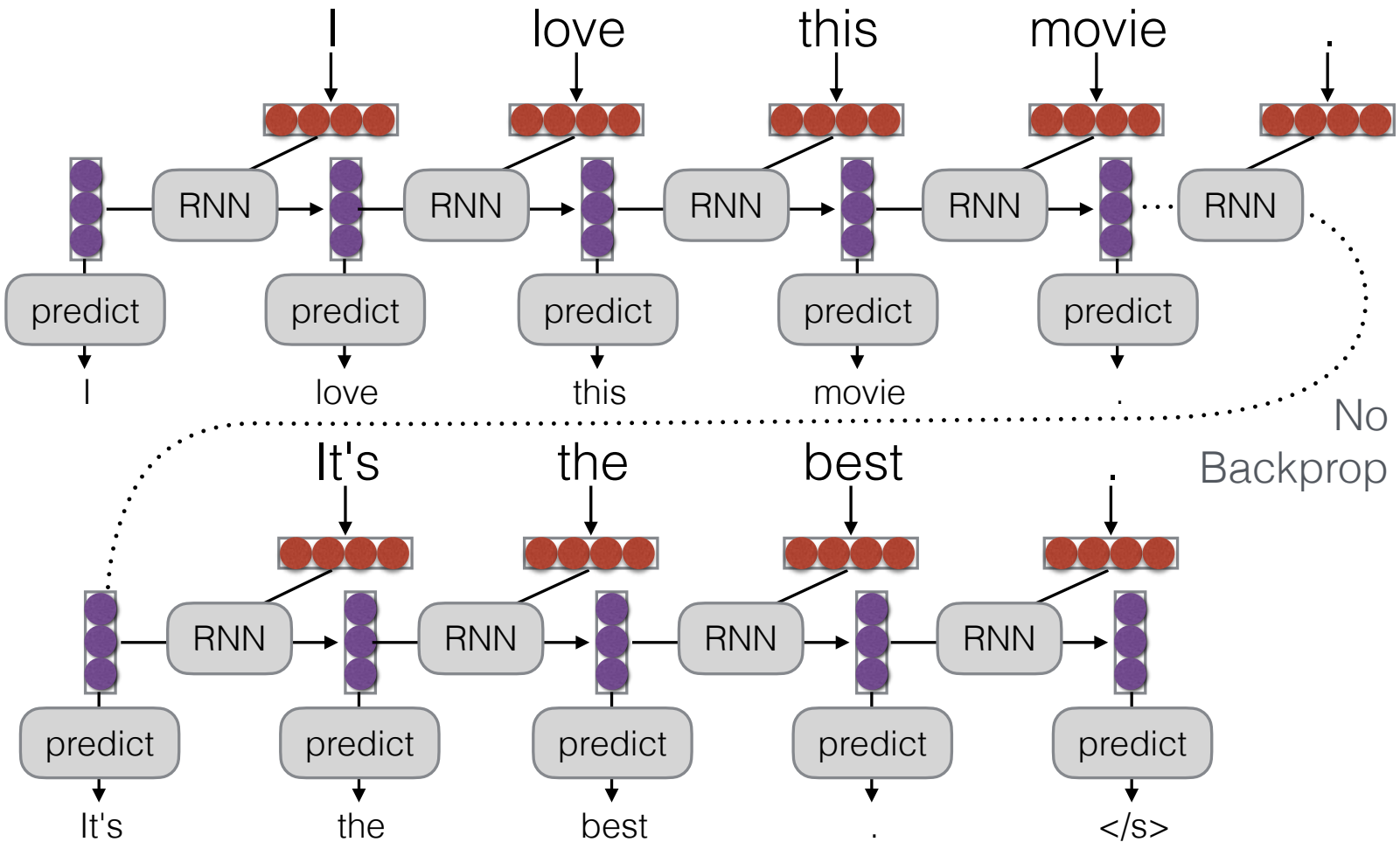
Low-rank Approximation

- Calculating the attention matrix is expensive, can it be predicted with a low-rank matrix?
- **Linformer:** Add low-rank linear projections into model (Wang et al. 2020)
- **Nystromformer:** Approximate using the Nystrom method, sampling "landmark" points (Xiong et al. 2021)



Tackling Complexity: Non-attentional Models

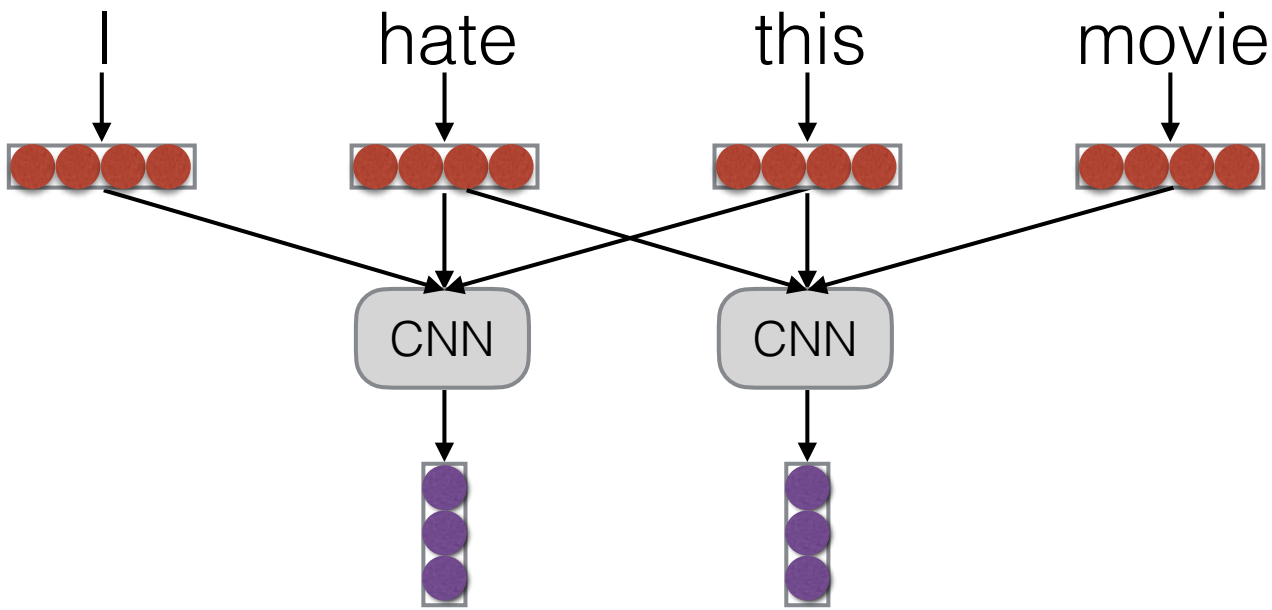
Reminder: RNNs



- Each RNN step depends on the previous - slow!

Convolution

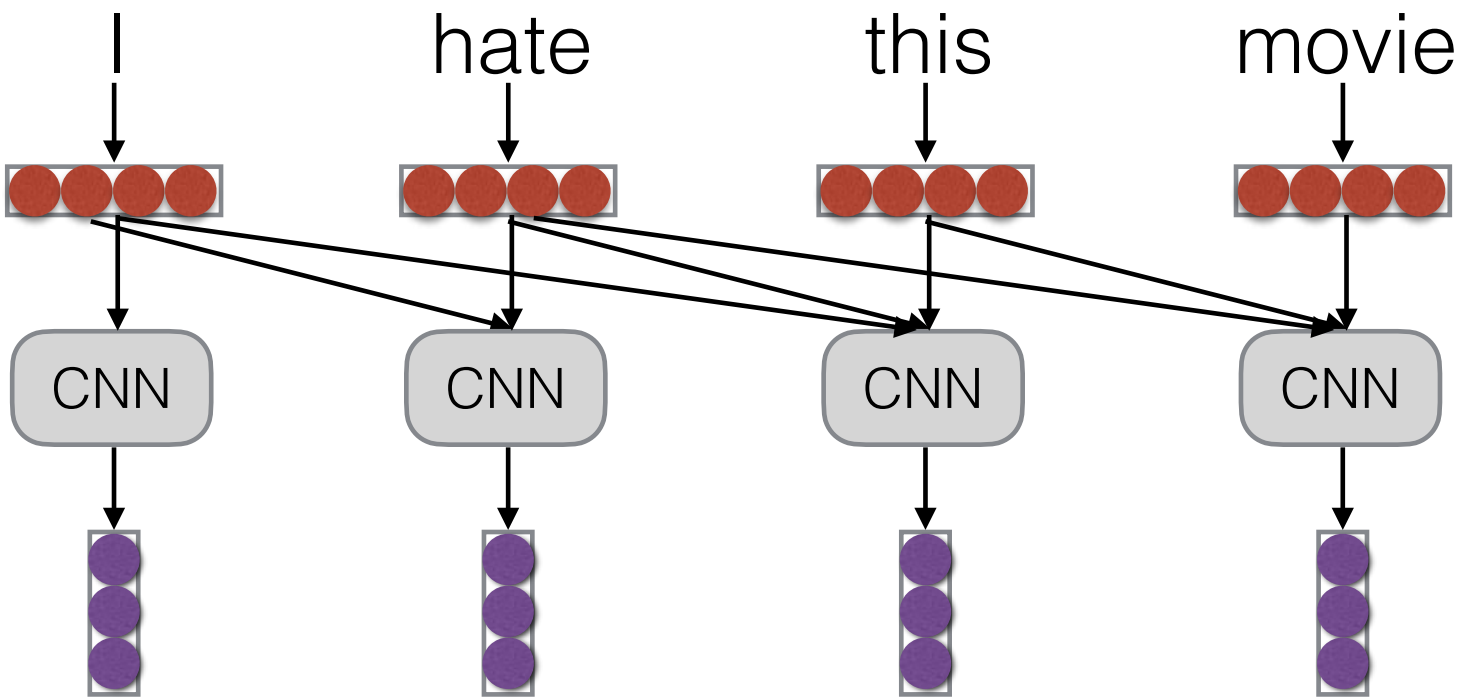
- Calculate based on local context



$$h_t = \mathrm{f}(W[x_{t-1}; x_t; x_{t+1}])$$

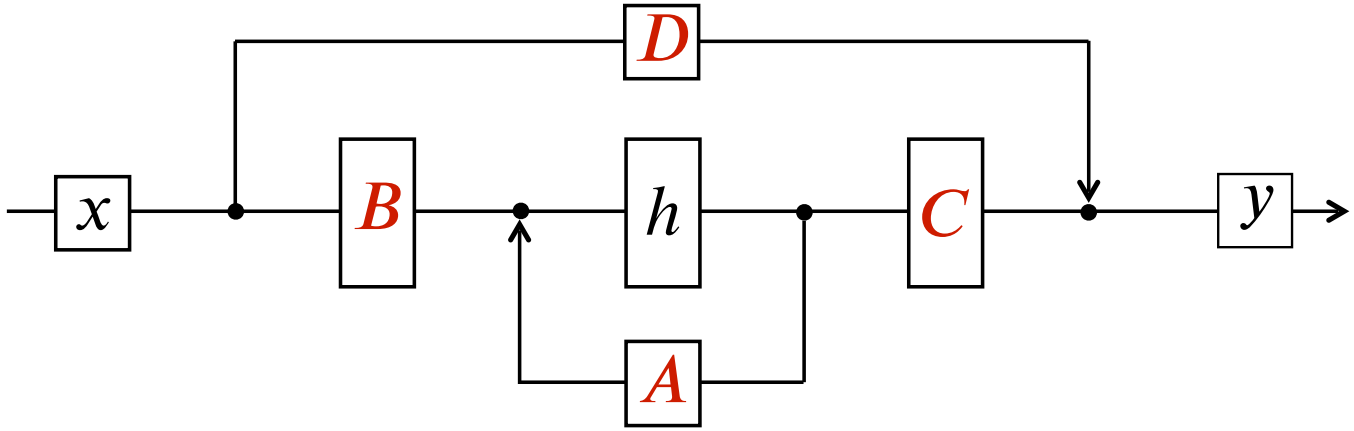
Convolution for Auto-regressive Models

- Functionally identical, just consider previous context



Structured State Space Models (Gu et al. 2021)

- Models that take a form like the following



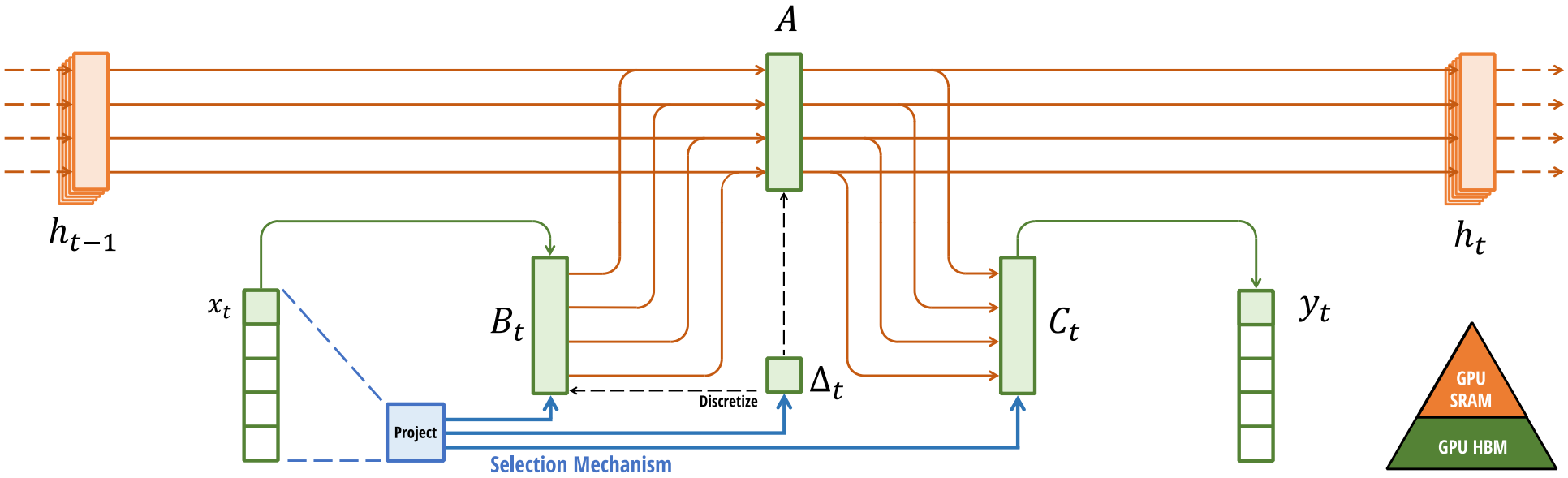
$$h'(t) = \boldsymbol{A}h(t) + \boldsymbol{B}x(t)$$
$$y(t) = \boldsymbol{C}h(t) + \boldsymbol{D}x(t)$$

- Because there are no non-linearities, the current h/x as a function of previous states can be calculated in advance

Slide Credit: Albert Gu

Selective State Space Models - Mamba
(Gu and Dao 2023)

- To improve modeling power of state space models, condition parameters on current input

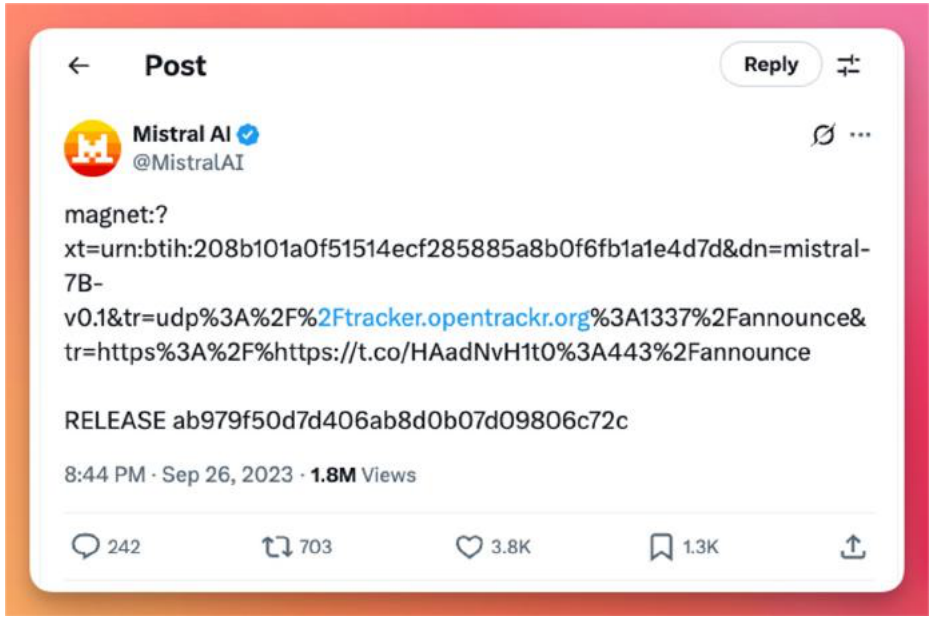


- Use efficient parts of GPU memory to handle expanded state

Open Model Landcape in 2025

Backstory: Open Models Pre-2025

Main players: Llama – dominant
Mistral – strong early start, fading
DeepSeek – interesting models
Qwen – growing similar to Llama



https://x.com/MistralAI/status/1706877320844509405?lang=en

2025 in open models

Major releases:

- January: DeepSeek R1
- February:
- March: Gemma 3
- April: Llama 4, Qwen 3
- June: MiniMax–M1, Baidu ERNIE 4.5
- July: Kimi K2, Qwen 3 Coder + more, GLM 4.5, StepFun Step3
- Aug: GPT-OSS, Nvidia Nemotron Nano, Seed-OSS
- Sep: Qwen Next, Qwen3 Omni, Qwen3 VL, GLM 4.6
- October:

Color code: **Chinese models**, Western models

Two modes of open model players

Many models as a platform: Llama, Qwen, Gemma, etc.

- Very strong researcher adoption
- Many model sizes (e.g. 1B, 3B, 7B, 30B, and 70B) and modalities

Single-path, high-quality models: DeepSeek, Z.ai, Moonshot/Kimi

- Few models that are often used heavily as substitutes for API frontier models

Other providers are in between, e.g. OpenAI's GPT-OSS and other Chinese labs.

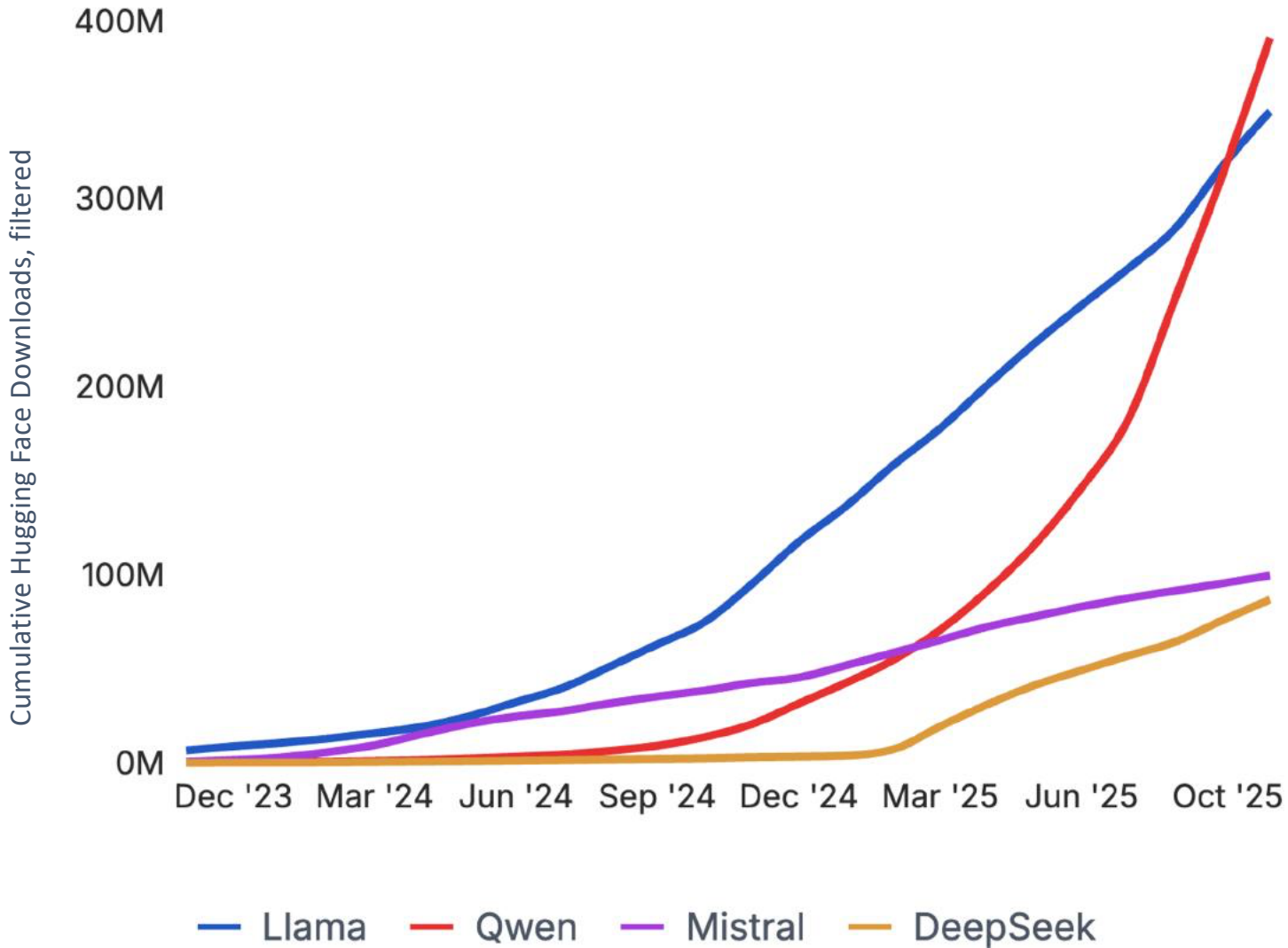
Qwen alone is roughly matching the entire American open model ecosystem today

Qwen's notable releases in 2025:

- **January:** Qwen2.5-1M (long context), Qwen2.5-VL (vision model), Qwen2.5-Max (API)
- **February:** QwQ-Max-Preview
- **March:** QwQ-32B (reasoning), Qwen2.5-VL-32B-Instruct, Qwen2.5-Omni, QVQ-Max (visual reasoning)
- **April:** Qwen3 (MoE + Dense open-weights)
- **June:** Qwen3 Embedding, Qwen VLo, Qwen-TTS
- **July:** Qwen3-Coder-480B-Agentic (Coding agent), Qwen-Image
- **August**: Qwen-Image-Edit, Qwen3-ASR-Flash
- **September**: Qwen3-Next, Qwen3-TTS-Flash, Qwen-Image-Edit-2509, Qwen3-Omni, Qwen3Guard, Qwen3-LiveTranslate-Flash, Qwen3-VL, Qwen3-Max

Qwen has surpassed Llama as the most used LLM family

Cumulative downloads of leading LLM organizations from key open-weight AI labs.



The ATOM Project.
https://atomproject.ai/

Next lecture:
Adapting LLMs