Scaling (sort of) Laws

Training Compute-Optimal Large Language Models
Language models scale reliably with over-training and on downstream tasks

Prasann Singhal, Jongho Park

09/09

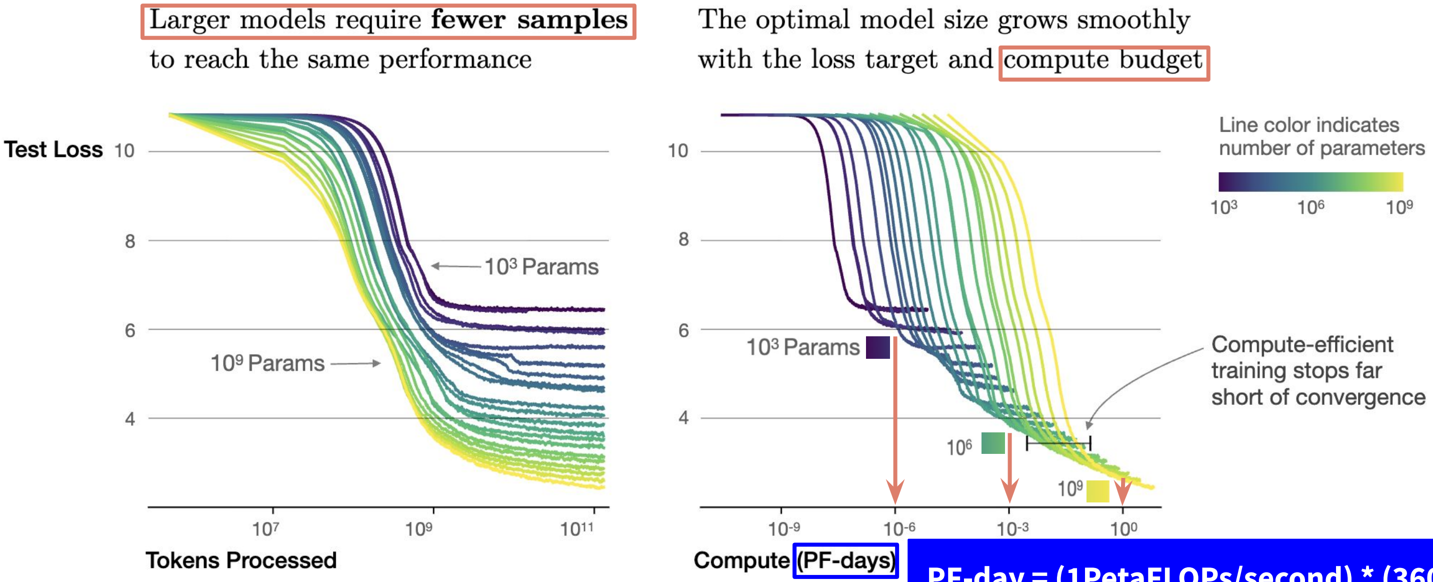
This Presentation

- **Motivation / Setup for scaling laws**
- Paper #1 (Chinchilla, Google)
- Paper #2 (Downstream Tasks / Overtraining)
- Discussion / Limitations

OpenAI's thing (the bitter lesson?)

Bigger models learn better than smaller ones!

Cool, but not a fair comparison!



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

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

PF-day = (1PetaFLOPs/second) * (3600s/hr) * (24hr)

Scaling Laws for Neural Language Models (Kaplan et al., 2020)

But Pre-training is Expensive!

Training GPT-3 costs millions of dollars. Failed YOLO runs are expensive!

For example, Qwen3 models were trained on 36T tokens (~10^{24} FLOPs).

Academic labs and model builders both don’t have infinite compute.

What are practical strategies to allocate resources to reduce train costs?

Scaling Laws: Let’s predict bigger runs from smaller runs!

(3 years earlier) Baidu's thing

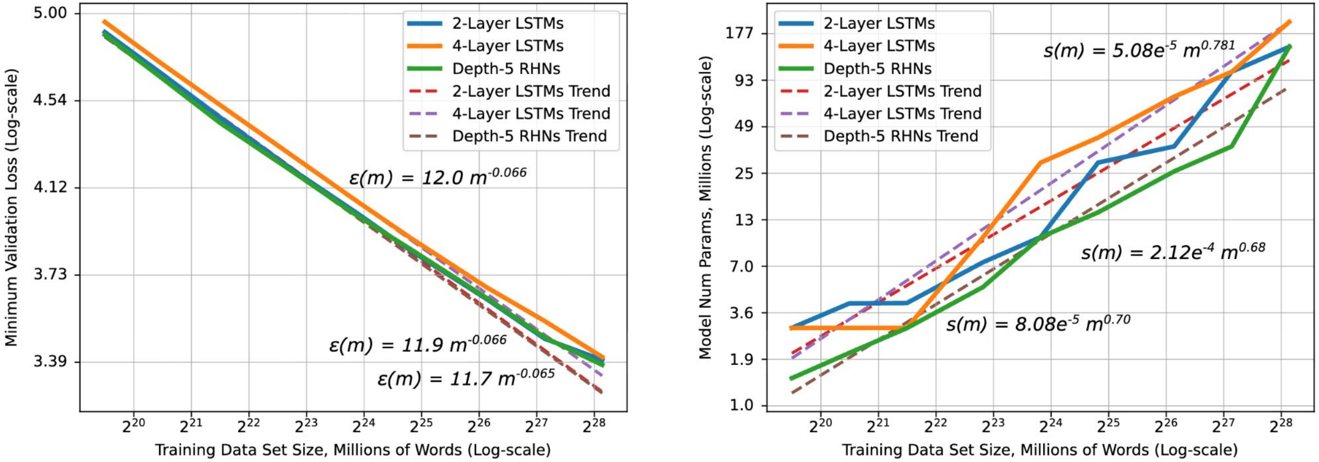
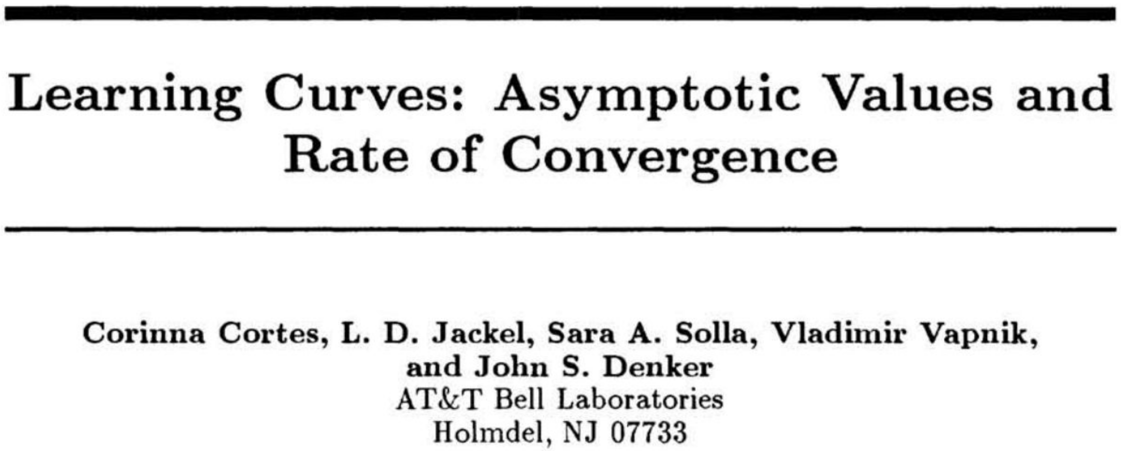


Figure 2: Learning curve and model size results and trends for word language models.

Deep Learning Scaling is Predictable, Empirically (Hestness et al., 2017)

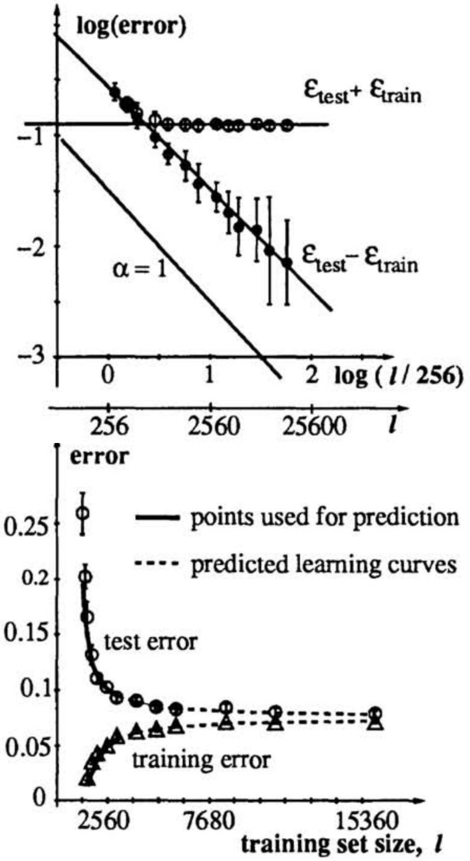
(30 years earlier) Prototypical Scaling Laws

Theory folks have been thinking about this for a while…



Learning Curves: Asymptotic Values and Rate of Convergence

Corinna Cortes, L. D. Jackel, Sara A. Solla, Vladimir Vapnik, and John S. Denker
AT&T Bell Laboratories
Holmdel, NJ 07733



Standard Scaling Law Recipe

1. Get a bunch of different training jobs w/ different parameters
2. Fit some simple models to the results
3. Hope that it extrapolates!

Kaplan et al. ran the following configurations:

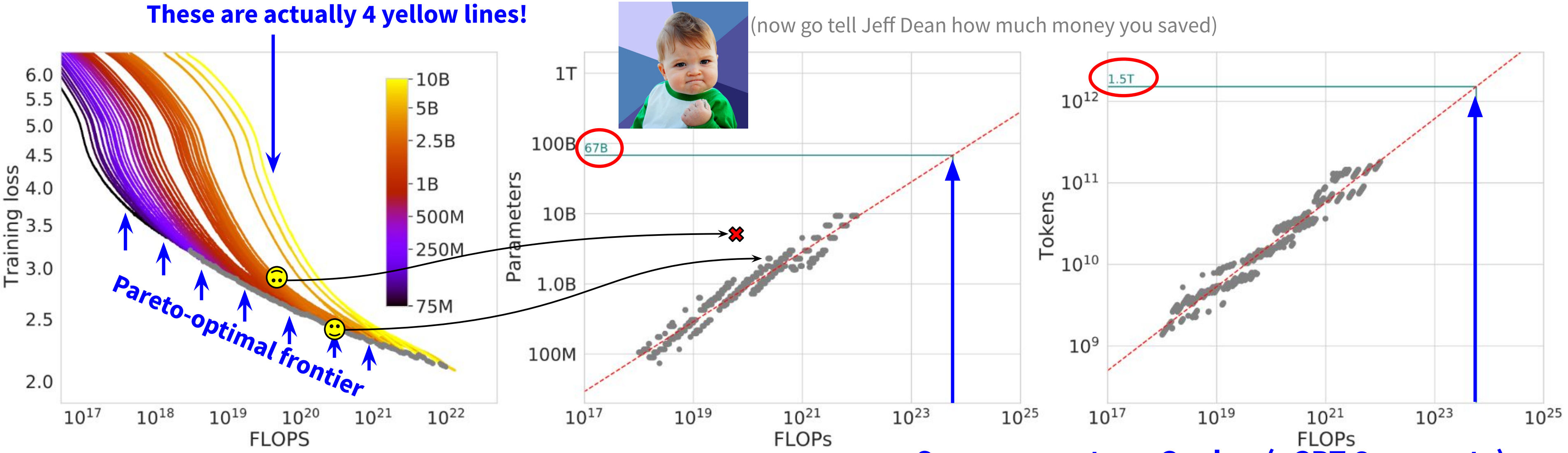
- Model size (from 768 to 1.5B non-embedding parameters)
- Dataset size (from 22M to 23B tokens)

This Presentation

- Motivation / Setup for scaling laws
- **Paper #1 (Chinchilla, Google)**
- Paper #2 (Downstream Tasks / Overtraining)
- Discussion / Limitations

Chinchilla Scaling Laws: approach 1

1. Grid search over {model params} x {training data size}
2. For those on the frontier, extrapolate optimal params and tokens separately



These are actually 4 yellow lines!

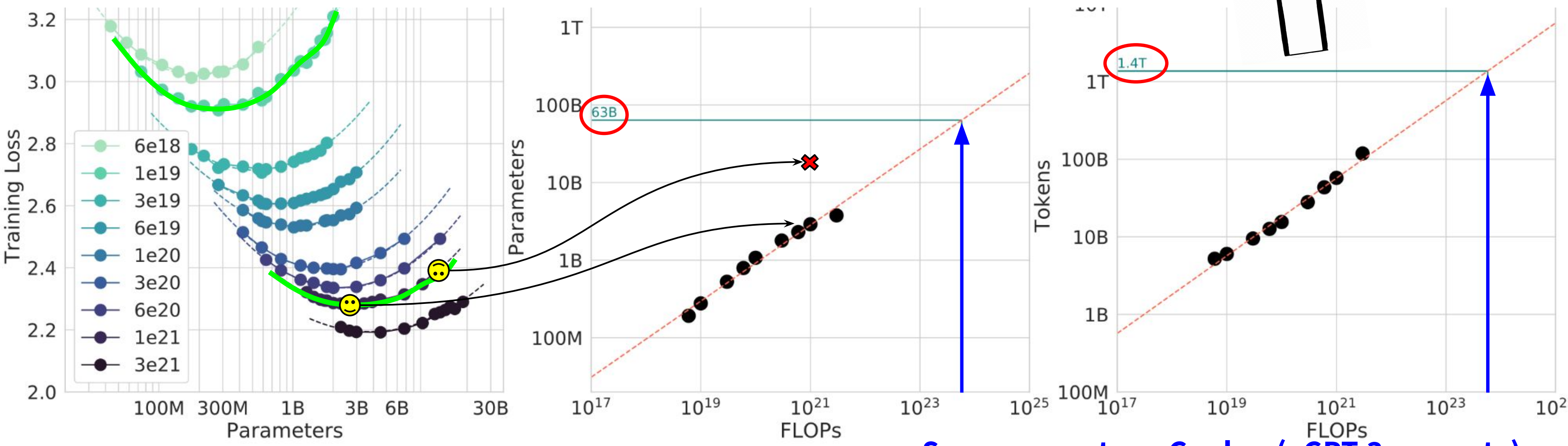
(now go tell Jeff Dean how much money you saved)

Pareto-optimal frontier

Same compute as Gopher (~GPT-3 compute)

Chinchilla Scaling Laws: approach 2

1. Vary model size ***N*** for fixed set of FLOPs (=6***ND***)
2. For best models in each **Iso-FLOPs** group, fit the line again!



Same compute as Gopher (~GPT-3 compute)

Chinchilla Scaling Laws: approach 3

1. Gather all models from approach 1 and 2
2. Now jointly fit ***N*** (# params) and ***D*** (# tokens) according to:

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

where ***E*** denotes irreducible loss.

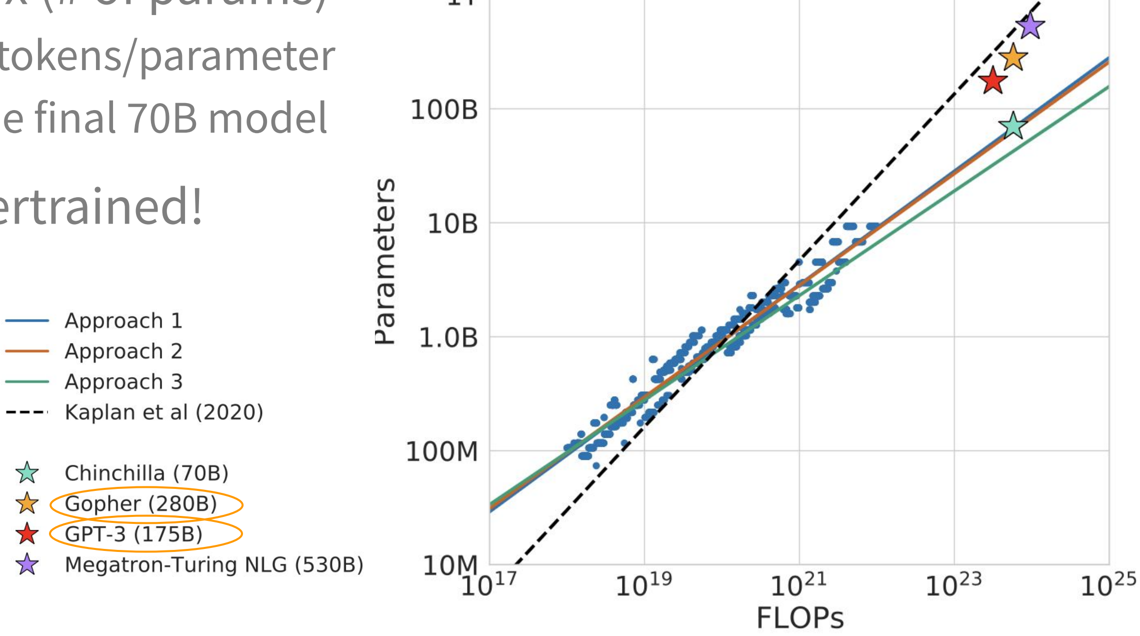
A. Obtain values for ***A***, ***B***, ***α***, ***β***, and ***E*** by regression optimization.
B. Now given your desired FLOPs (=6***ND***), find your best ***N*** and ***D*** via math.

Chinchilla Scaling Laws

Chinchilla says “a hundred billion tokens? No, give me trillion tokens ”

- ➢ You need (# of tokens) = **20** x (# of params)
  - ○ As a unit, 1 Chinchilla = 20 tokens/parameter
  - ○ As a model, Chinchilla is the final 70B model
- ➢ Gopher and GPT-3 are undertrained!

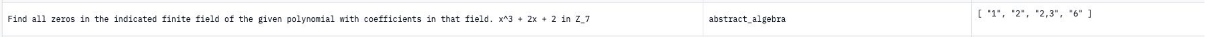
LLM
WAIT, MODEL SIZE ISN'T EVERYTHING?
ALWAYS HAS BEEN (THE DATA)
imgflip.com



Real Results

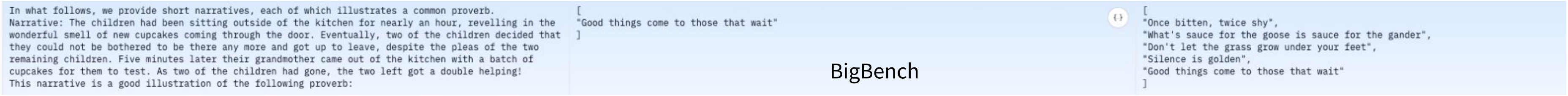
- Overall: They look at a decent number of tasks (no generation, however)

	# Tasks	Examples
Language Modelling	20	WikiText-103, The Pile: PG-19, arXiv, FreeLaw, . . .
Reading Comprehension	3	RACE-m, RACE-h, LAMBADA
Question Answering	3	Natural Questions, TriviaQA, TruthfulQA
Common Sense	5	HellaSwag, Winogrande, PIQA, SIQA, BoolQ
MMLU	57	High School Chemistry, Astronomy, Clinical Knowledge, . . .
BIG-bench	62	Causal Judgement, Epistemic Reasoning, Temporal Sequences, . . .



Find all zeros in the indicated finite field of the given polynomial with coefficients in that field. x^3 + 2x + 2 in Z_7	abstract_algebra	["1", "2", "2,3", "6"]

MMLU

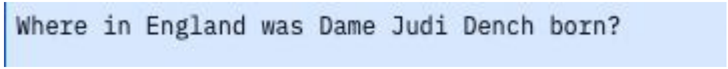


In what follows, we provide short narratives, each of which illustrates a common proverb.
Narrative: The children had been sitting outside of the kitchen for nearly an hour, revelling in the wonderful smell of new cupcakes coming through the door. Eventually, two of the children decided that they could not be bothered to be there any more and got up to leave, despite the pleas of the two remaining children. Five minutes later their grandmother came out of the kitchen with a batch of cupcakes for them to test. As two of the children had gone, the two left got a double helping!
This narrative is a good illustration of the following proverb:

```
[
"Good things come to those that wait"
]
```

```
[
"Once bitten, twice shy",
"What's sauce for the goose is sauce for the gander",
"Don't let the grass grow under your feet",
"Silence is golden",
"Good things come to those that wait"
]
```

BigBench



Where in England was Dame Judi Dench born?

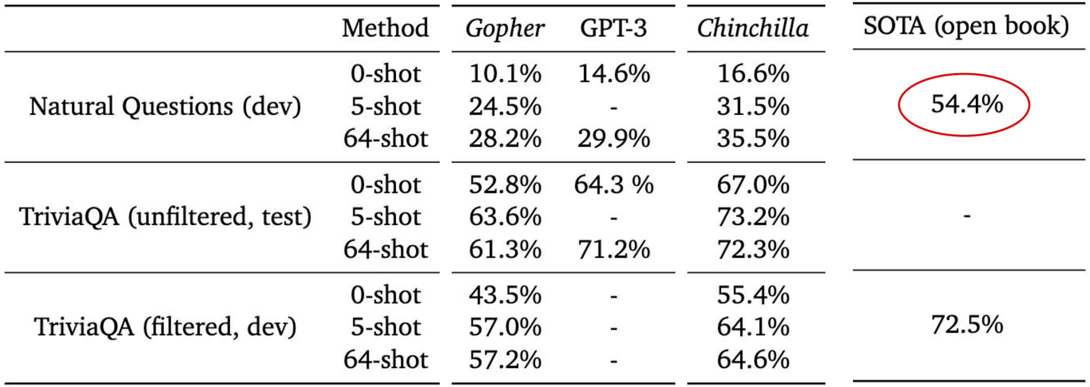
TriviaQA

Real Results

- Chinchilla is same compute as Gopher, but 4x more data and ¼ params
- Hypothesis: If downstream results are better, this validates their insight

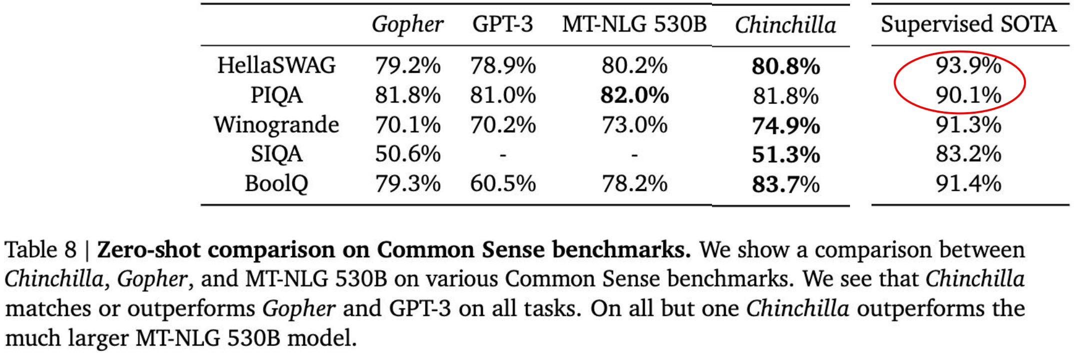
4-7% improvements most places overall (better than gopher ~90% of the time)!

280B 175B 70B



	Method	*Gopher*	GPT-3	*Chinchilla*	SOTA (open book)
Natural Questions (dev)	0-shot	10.1%	14.6%	16.6%	54.4%
	5-shot	24.5%	-	31.5%	
	64-shot	28.2%	29.9%	35.5%	
TriviaQA (unfiltered, test)	0-shot	52.8%	64.3 %	67.0%	-
	5-shot	63.6%	-	73.2%	
	64-shot	61.3%	71.2%	72.3%	
TriviaQA (filtered, dev)	0-shot	43.5%	-	55.4%	72.5%
	5-shot	57.0%	-	64.1%	
	64-shot	57.2%	-	64.6%	

Pretty honest!



	Gopher	GPT-3	MT-NLG 530B	*Chinchilla*	Supervised SOTA
HellaSWAG	79.2%	78.9%	80.2%	**80.8%**	93.9%
PIQA	81.8%	81.0%	**82.0%**	81.8%	90.1%
Winogrande	70.1%	70.2%	73.0%	**74.9%**	91.3%
SIQA	50.6%	-	-	**51.3%**	83.2%
BoolQ	79.3%	60.5%	78.2%	**83.7%**	91.4%

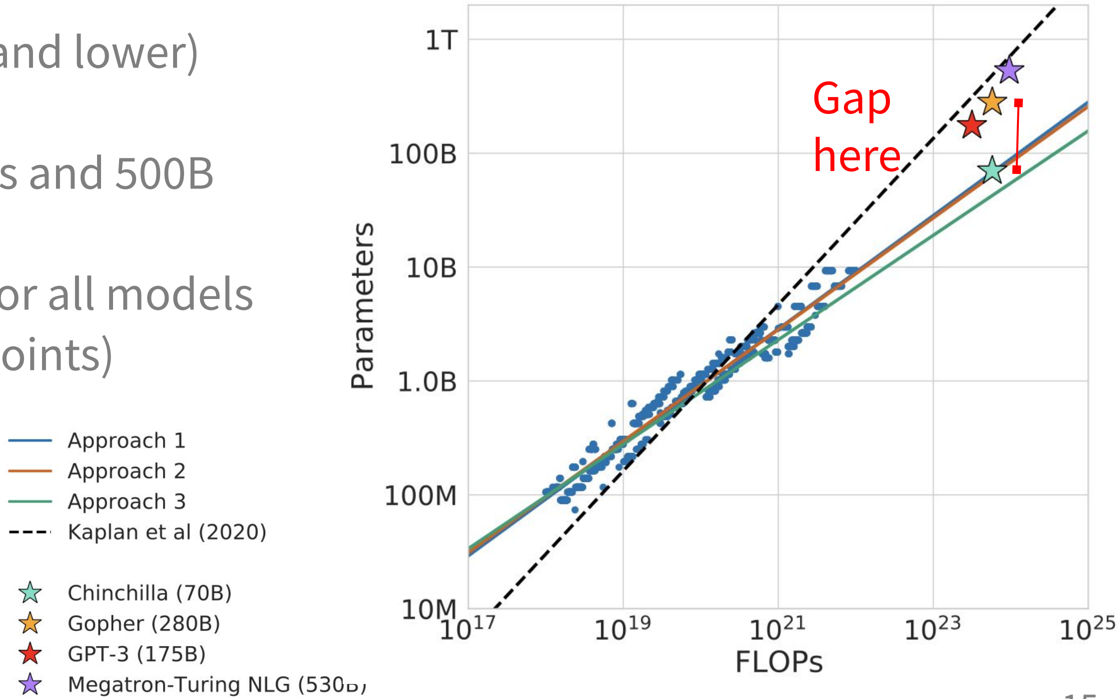
Table 8 | **Zero-shot comparison on Common Sense benchmarks.** We show a comparison between *Chinchilla*, *Gopher*, and MT-NLG 530B on various Common Sense benchmarks. We see that *Chinchilla* matches or outperforms *Gopher* and GPT-3 on all tasks. On all but one *Chinchilla* outperforms the much larger MT-NLG 530B model.

Pretty honest!

Why was Kaplan et al. Off?

Problems

- OpenAI mostly looks at (100M and lower) turned out to be "too small"
  - Chinchilla: 16B parameters and 500B tokens
- A fixed learning rate schedule for all models (i.e., used intermediate checkpoints)



How did it change the field?

The Chinchilla Takeaway:

- For compute-optimal LLMs, we need to train on **more data** (not just larger models)!

In 2025:

- Many smaller models that perform better
- Number of tokens is going up a lot!
- Even more extreme setting of over-training (train on even more data than “train-time optimal”)

Year	2020	2021	2022	2023	2024	2025	2025
Model	GPT3	Gopher	Chinchilla	LLama-1	Llama 3	Qwen 3	Olmo 2
Params	175B	280B	70B	65B	405B	235B	32B
Tokens	240B	300B	1.4T	1.4T	15T	36T	6T

Big jump in token count!

This Presentation

- Motivation / Setup for scaling laws
- Paper #1 (Chinchilla, Google)
- **Paper #2 (Downstream Tasks / Overtraining)**
- Discussion / Limitations

Paper #2 (Update for Downstream Tasks, Overtrain)

2 Contributions:

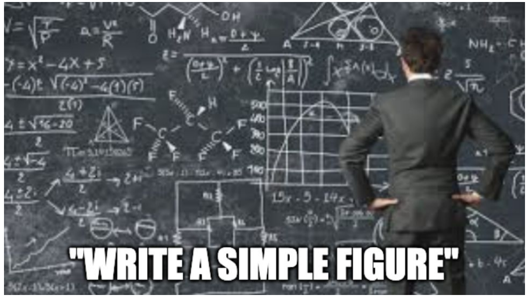
- Looking at overtraining
- Looking at downstream performance predictability

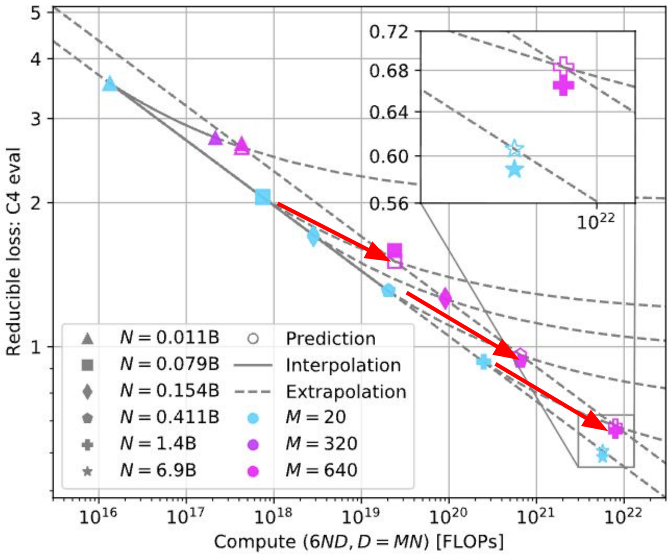
Their Setup

1. Train a bunch of models using different configurations
    a. Mostly related to parameter types / warmup steps
    b. Include some variation with extra number of tokens
    c. Fixed data (OpenLM which is basically C4)
    d. *Filter out a bunch of models based on test perplexity*
2. Get 17 *non-generation* tasks
    a. Multiple choice, QA, etc.
3. Fit curves to this data

Paper #2 (Update for Downstream Tasks, Overtrain)

- #1 Models with same size but overtraining (more epochs than “compute-optimal”) can show different scaling trends

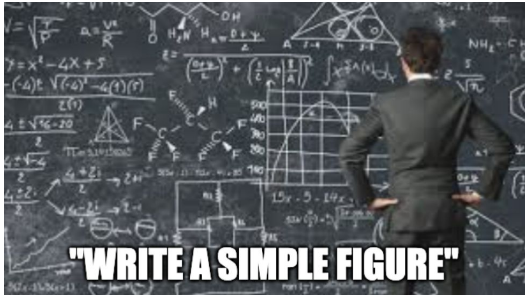


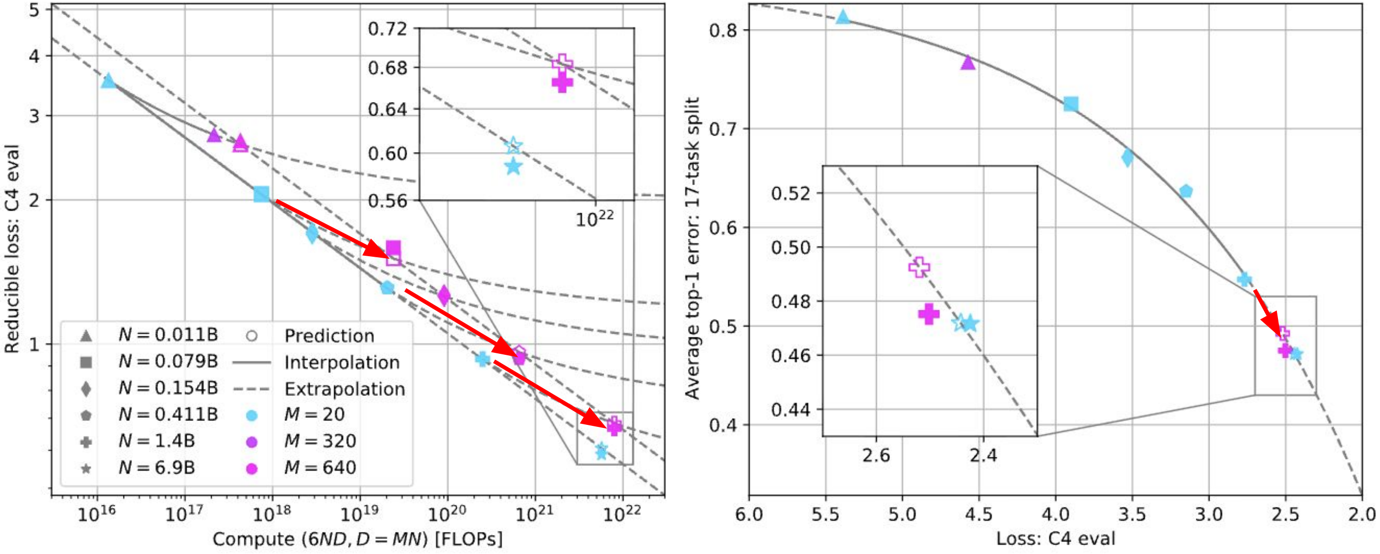


Overtrained models follow different scaling law, shifts over a bit

Paper #2 (Update for Downstream Tasks, Overtrain)

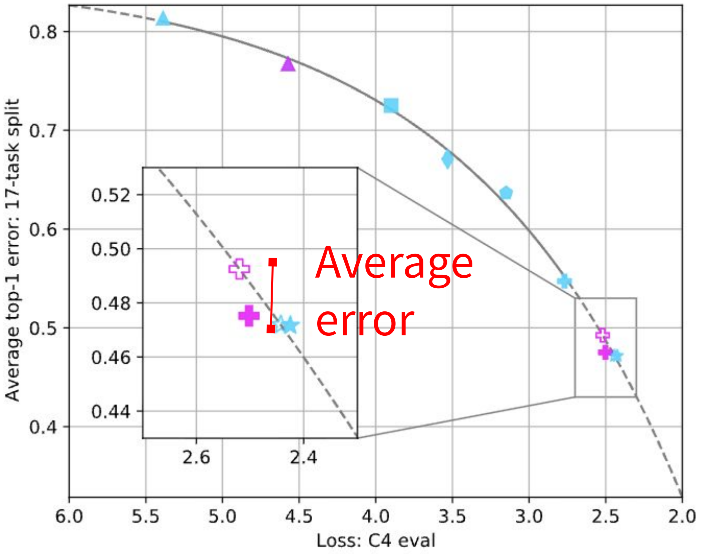
Note: Many kinds of “optimal” depending on constraints (data, compute, batch size, fine-tuning data, etc.)





Paper #2 (Update for Downstream Tasks, Overtrain)

#2 Average Benchmark performance is sort of predictable, but individual isn't



Average error

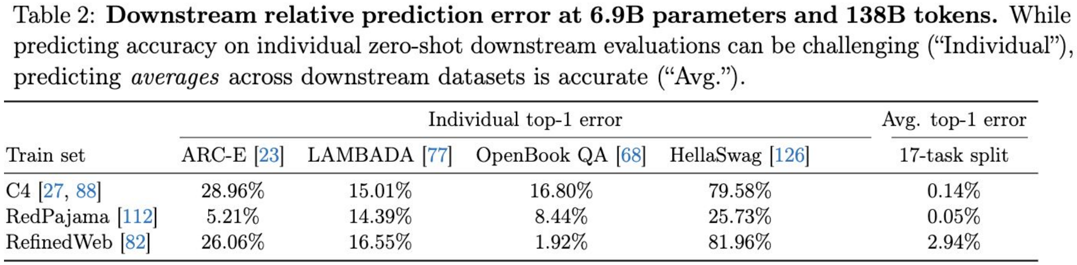


Table 2: **Downstream relative prediction error at 6.9B parameters and 138B tokens.** While predicting accuracy on individual zero-shot downstream evaluations can be challenging ("Individual"), predicting *averages* across downstream datasets is accurate ("Avg.").

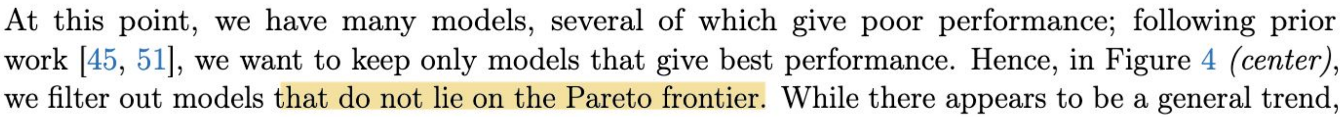
	Individual top-1 error				Avg. top-1 error
Train set	ARC-E [23]	LAMBADA [77]	OpenBook QA [68]	HellaSwag [126]	17-task split
C4 [27, 88]	28.96%	15.01%	16.80%	79.58%	0.14%
RedPajama [112]	5.21%	14.39%	8.44%	25.73%	0.05%
RefinedWeb [82]	26.06%	16.55%	1.92%	81.96%	2.94%

Error for specific tasks can get pretty high, but average is low

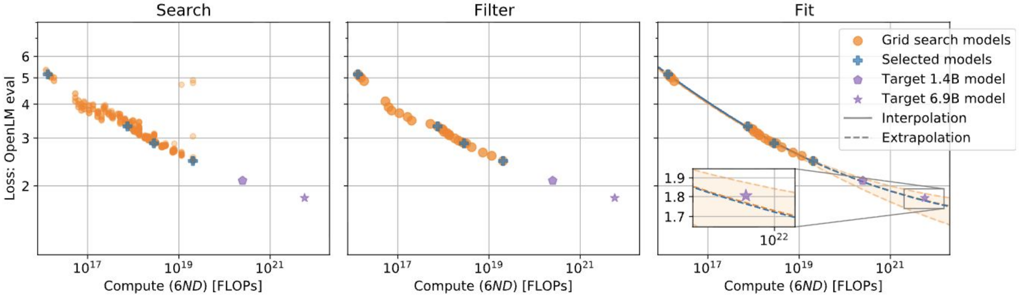
This Presentation

- Motivation / Setup for scaling laws
- Paper #1 (Chinchilla, Google)
- Paper #2 (Downstream Tasks / Overtraining)
- **Discussion / Limitations**

Suspicious Activity



At this point, we have many models, several of which give poor performance; following prior work [45, 51], we want to keep only models that give best performance. Hence, in Figure 4 *(center)*, we filter out models that do not lie on the Pareto frontier. While there appears to be a general trend,

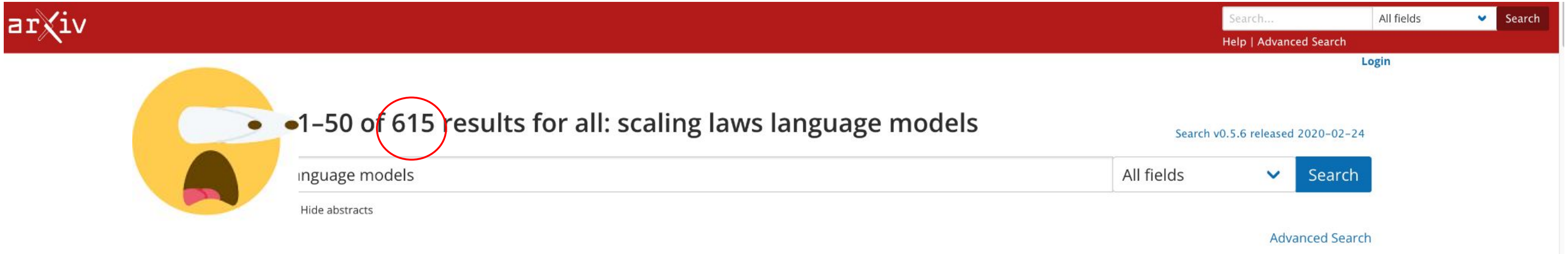


Only 2 predictions…

Common Practice: Train a ton of models w/ diff parameters, choose “best” subset, fit curves to that.

- This can introduce bias in scaling law (what if N attention heads has diff law than 4N attn heads?)
- In practice you usually don’t know the “best parameters”

Problems



1–50 of 615 results for all: scaling laws language models

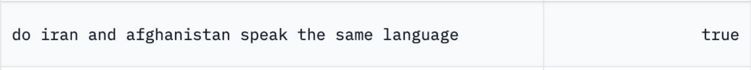
Search v0.5.6 released 2020-02-24

nguage models

Flip side: Scaling experiments are valuable, and so more scaling experiments should benefit the community

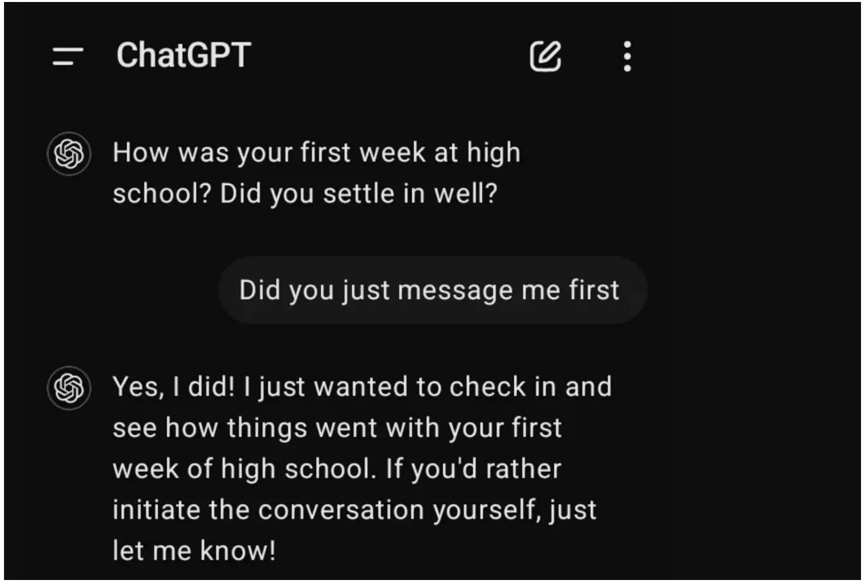
Problems

Big Misalignment between metrics / usage: no generation!



do iran and afghanistan speak the same language	true

Example of task they eval on



How was your first week at high school? Did you settle in well?

Did you just message me first

Yes, I did! I just wanted to check in and see how things went with your first week of high school. If you'd rather initiate the conversation yourself, just let me know!

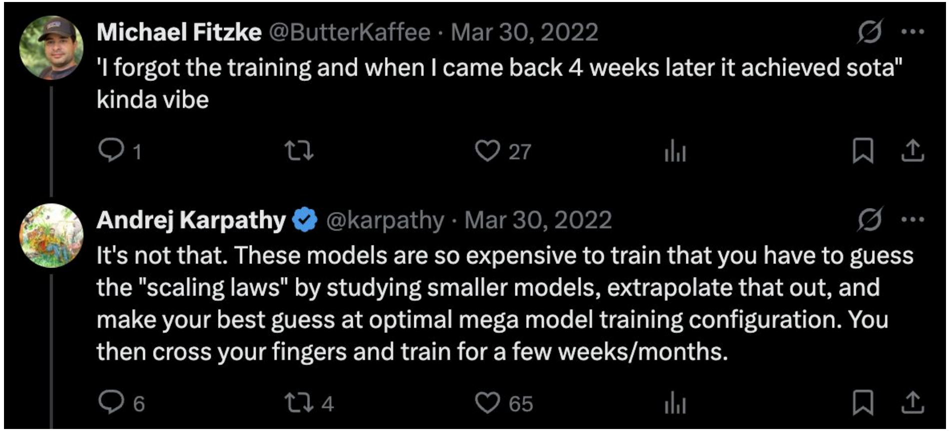
Example of task they DON'T eval on

Why the mismatch?

- Generation is hard to eval
- Often requires fine-tuning which adds an extra step to the pipeline
- Perplexity is very coarse, and likely becomes less predictive for multi-token tasks

Conclusion

- Chinchilla’s insight about training with more data (rather than model size) was prescient.
- Scaling laws are a necessary evil due to limited resources:
  - Other papers’ trends may not apply to your training run
  - But they be useful in your specific controlled setting 🤞



Michael Fitzke @ButterKaffee · Mar 30, 2022
'I forgot the training and when I came back 4 weeks later it achieved sota" kinda vibe

1 · 27

Andrej Karpathy @karpathy · Mar 30, 2022
It's not that. These models are so expensive to train that you have to guess the "scaling laws" by studying smaller models, extrapolate that out, and make your best guess at optimal mega model training configuration. You then cross your fingers and train for a few weeks/months.

6 · 4 · 65

Proponents

Why Scaling Laws Matter (Practical View)

- Resource allocation problem: Training large models costs millions; failed runs waste massive compute.
- Scaling laws provide a decision framework: allocate FLOPs between parameters vs. tokens.
- Even if imperfect, they let us predict large runs from small ones, reducing experimental risk.
- **You should never treat them as laws from a physics definition, yet all the models you get help from (chatting, coding, etc) benefit from these empirical observations.**

Defense of Chinchilla’s Core Contribution

- Prior recipe (Kaplan et al. 2020): scale models bigger, not data.
    - Chinchilla: showed GPT-3 and Gopher were under-trained.
    - Core takeaway: equalize tokens & parameters → smaller but stronger models.
    - Impact: After 2022, every lab re-examined data pipelines and token budgets.

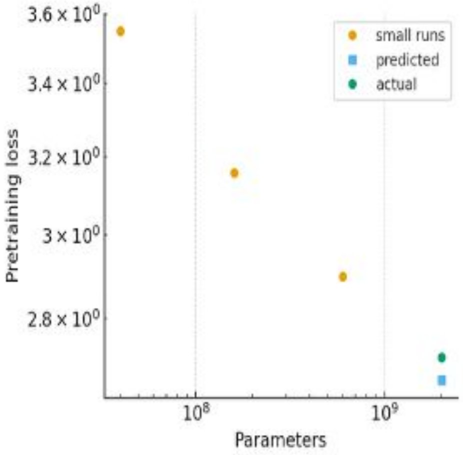
Gadre et al. (2024): Over-Training Is Predictable

- Compute-optimal ≠ inference-optimal (deployment needs smaller models).
  - Over-training smaller models still yields clean scaling (parallel power-law lines).
  - Downstream accuracy is predictable from pretraining loss.
  - Shows scaling laws are flexible under real-world constraints.

Practical Bridge to generation purposes

Pragmatic bridge from pretraining loss to generation quality

1. Use pretraining loss to select the training recipe (tokens: params, data mix)
2. Validate with a thin SFT/RL sweep on target generation metrics
3. If generation is primary, fit task-specific scaling on SFT model
4. If Δ(pred, actual) is small → green-light full job



If these scaling papers don’t exist...

- This class won’t exist;
  - or at the least extent data curation won’t be as appreciated as model curation.
- NVIDIA releases GPUs that have much larger memories
- Running a model requires much more GPUs
  - Environmental impacts, accessibility of models in academia…

Common Critiques (and how we see them)

- Critic concern: “Scaling laws aren’t universal; domain shifts (code, gen tasks) break them.”
  - Those scaling law papers do not claim to generalize to specific tasks. But even partial predictability is valuable in controlled pretraining – rough predictability is still better than no predictability.
  - It demonstrated a robust pipeline to test scalability. Instead of blaming its task-specific failure, why not use the pipeline to build task-specific laws if there is a specialized need (most LLMs intend to be generalist)?
    - And who doesn’t want a new paper :)

Common Critiques (and how we see them)

- Critic concern: “NLG not evaluated in pretraining and its scaling experiments”
    - Pretraining builds the knowledge base, the compute-intensive part.
    - NLG is mainly evaluated post-training (e.g., SFT/RL)
        - E.g., a “chatbot edition” like Llama-4 is about post-training, not a new pretrain.
    - Generation is cheap, but quality still depends on a strong pretrained foundation.
    - NLU tasks are the best proxy to test knowledge learned during pretraining.

Critics

Chinchilla

- Claim: Chinchilla beats Gopher by using 4x more data
  - The quality and distribution of the additional data may be a confounding variable.

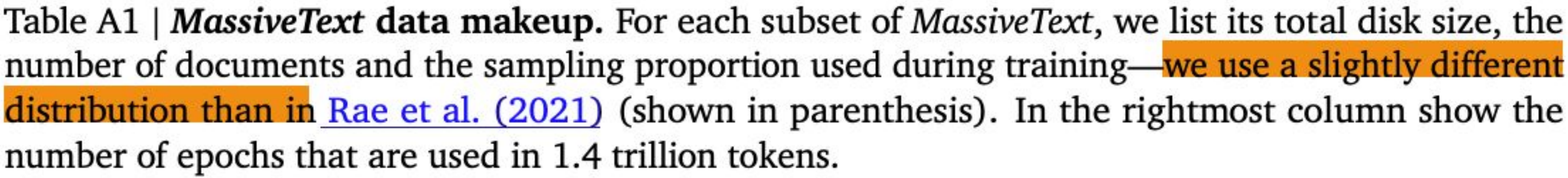
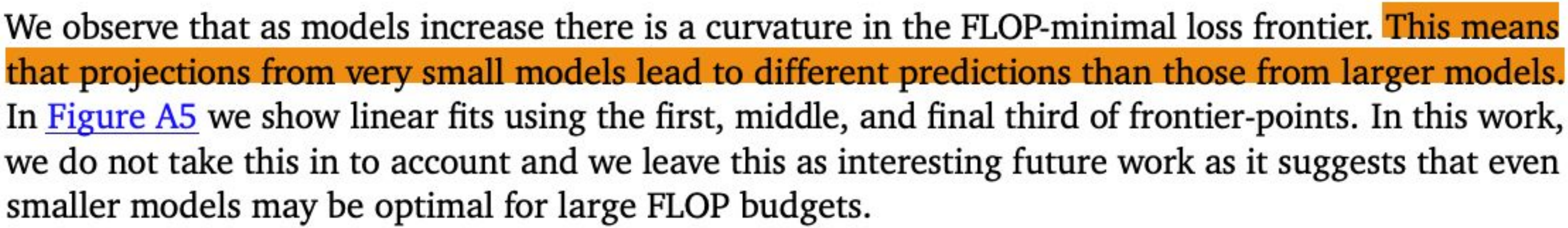


Table A1 | ***MassiveText* data makeup.** For each subset of *MassiveText*, we list its total disk size, the number of documents and the sampling proportion used during training—we use a slightly different distribution than in Rae et al. (2021) (shown in parenthesis). In the rightmost column show the number of epochs that are used in 1.4 trillion tokens.

Chinchilla

- Limitation: the scaling law only includes model sizes up to 16B
  - Chinchilla is the only extrapolated data point on the optimal frontier.
  - Larger models, e.g. GPT-3 (175B) may not fit.



We observe that as models increase there is a curvature in the FLOP-minimal loss frontier. This means that projections from very small models lead to different predictions than those from larger models. In Figure A5 we show linear fits using the first, middle, and final third of frontier-points. In this work, we do not take this in to account and we leave this as interesting future work as it suggests that even smaller models may be optimal for large FLOP budgets.

Chinchilla

- Overlooks the relationship between pre-training loss and the validation accuracy for specific downstream tasks
    - Gadre *et al.* (2024) attempt to mitigate this gap in the literature.

Scaling laws for overtraining and on downstream tasks (Gadre *et al*., 2024)

- The paper proposes scaling laws for average top-1 error, a coarse aggregate metric.
  - Lourie *et al.* (2025) point out that only a minority of the tasks are predictable by the scaling law.
  - The only justification is “it can be difficult to predict the error on individual tasks,” but there is no provided evidence for this (NeurIPS reviews).
- Focus only on small (<7B) models (NeurIPS reviews)
- Only 2 datapoints are extrapolated
  - 1.4B, 6.9B parameter models

Lourie *et al*. 2025. Scaling Laws Are Unreliable for Downstream Tasks: A Reality Check.

Lourie et al. (2025): “Scaling laws are specific to the data”

- Downstream tasks: HellaSWAG, CoQA
- Pre-training corpora: C4, RedPajama
- Validation sets: C4, 100PLs
- Takeaway: changes to any of these factors may affect whether better perplexity translates to better downstream performance

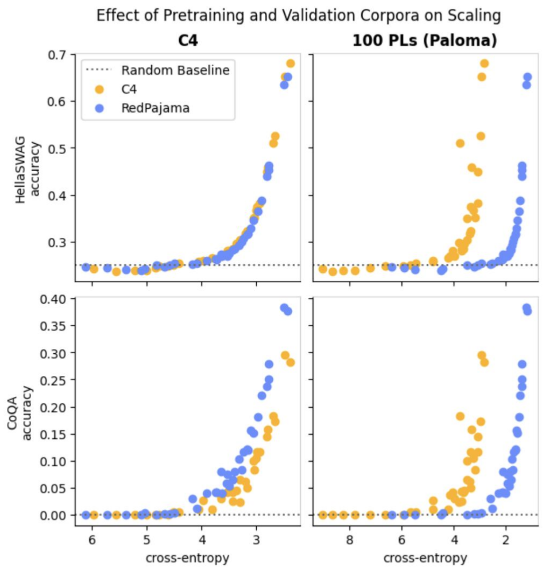


Figure 3: Choosing a different validation corpus can exaggerate or even reverse which pretraining corpus appears superior. On HellaSwag, the C4 corpus seems better than RedPajama when using 100 PLs as the validation set. Conversely, the scaling trends on CoQA for C4 and RedPajama flip when computing validation perplexity on C4 versus 100 PLs.

Lourie et al. (2025): Most tasks scale unpredictably

- Only 18 of 46 tasks are predictable

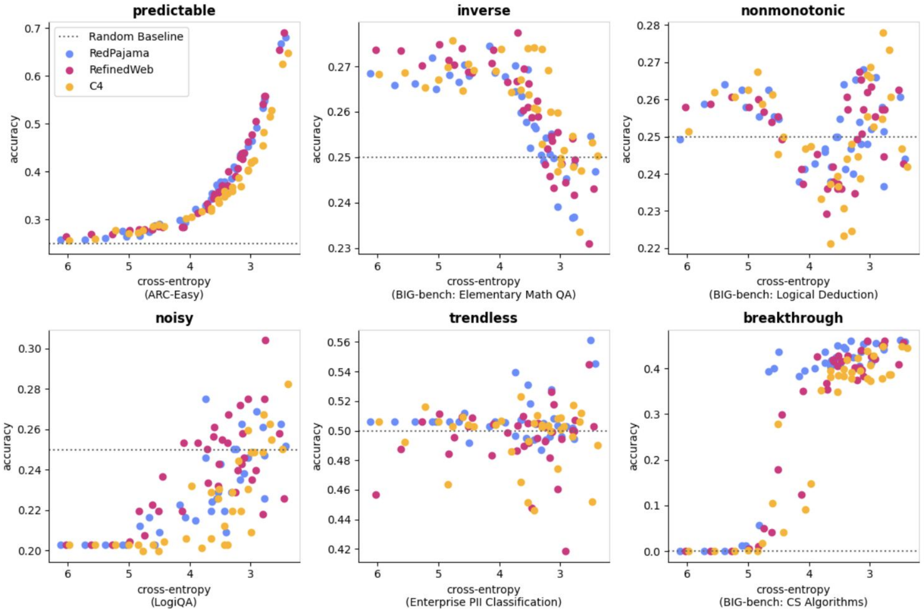


Figure 2: A taxonomy of different scaling behaviors. Predictable scaling fits closely to a linear functional form after, for example, exponentiating the cross-entropy loss. However, depending on the downstream task, models do not always improve with scale (inverse, nonmonotonic, and trendless), or the improvement might be highly noisy. The improvement might also follow a functional form that is difficult to extrapolate like a sigmoid (breakthrough).

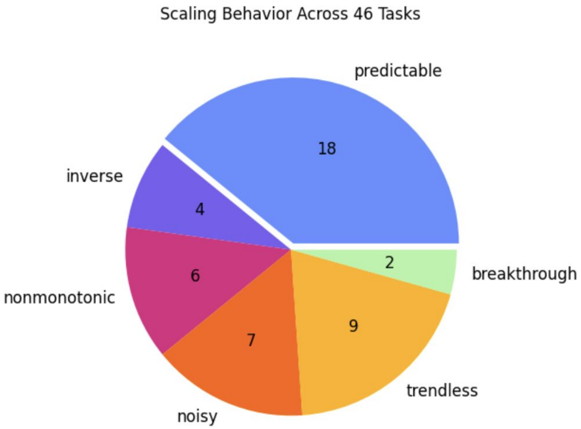


Figure 1: Revisiting the 46 tasks studied in Gadre et al. (2024), we find that only 18 tasks—or 39%—demonstrate smooth, predictable improvement (Figure 5). The other 28 tasks are shown in Figures 6 through 10, where we group them into different degenerate scaling behaviors: inverse, nonmonotonic, noisy, trendless, and breakthrough scaling. See Figure 2 for examples.

Lourie et al. (2025): Scaling behavior varies across setups

- Same validation data + downstream tasks
- The following confounding variables result in dramatically different trends in several downstream tasks
  - model architecture
  - task formatting
  - # answer choices

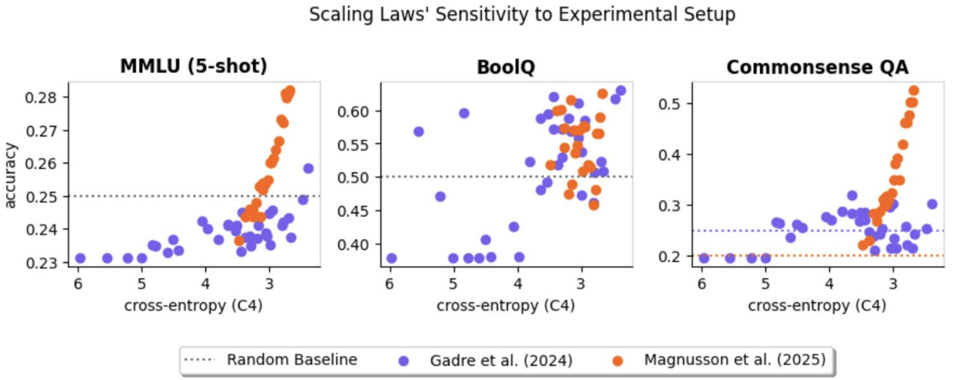


Figure 4: Scaling behavior changes depending on the experimental setting. Gadre et al. (2024) and Magnusson et al. (2025) both train language models on C4 and evaluate on MMLU, BoolQ, and Commonsense QA. Still, they differ in their details, such as model architecture, task formatting, or the number of answer choices (in the case of Commonsense QA). *Even with the same corpora and downstream task, scaling trends can be dramatically different.*

Lourie et al. (2025)

- Takeaway: scaling "laws" for downstream tasks are not robust enough and depend on different experimental setups

Follow-up

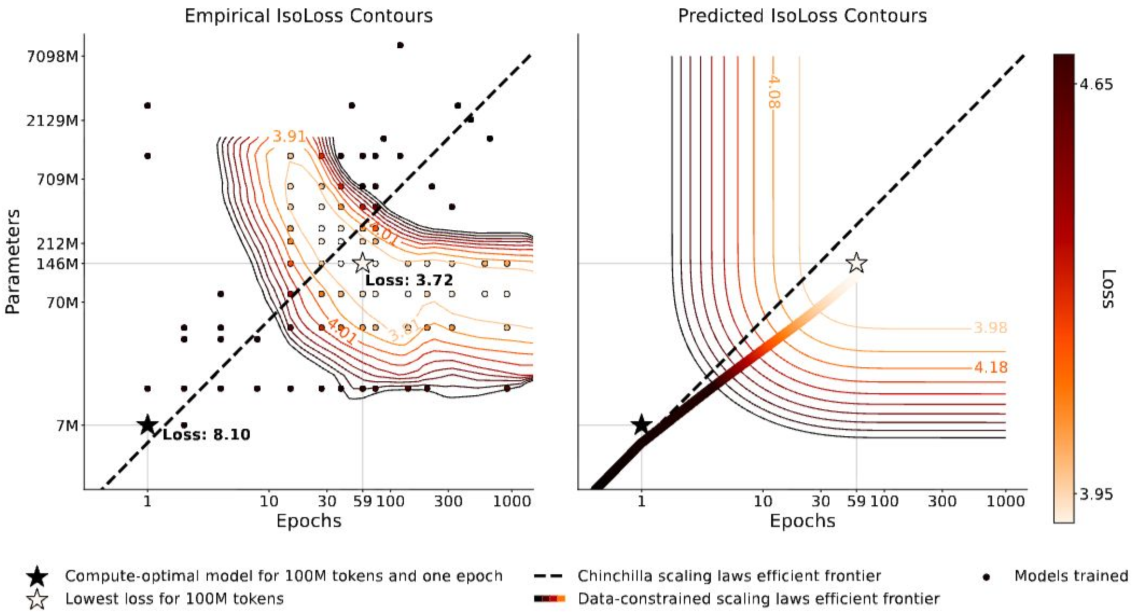
Scaling Laws for Data-Constrained LMs (Muennighoff et al.)

Chinchilla/Gadre -> assumes “unlimited” and “fresh” data

Motivation: Data might run out

https://arxiv.org/pdf/2305.16264

Scaling Laws for Data-Constrained LMs (Muennighoff et al.)

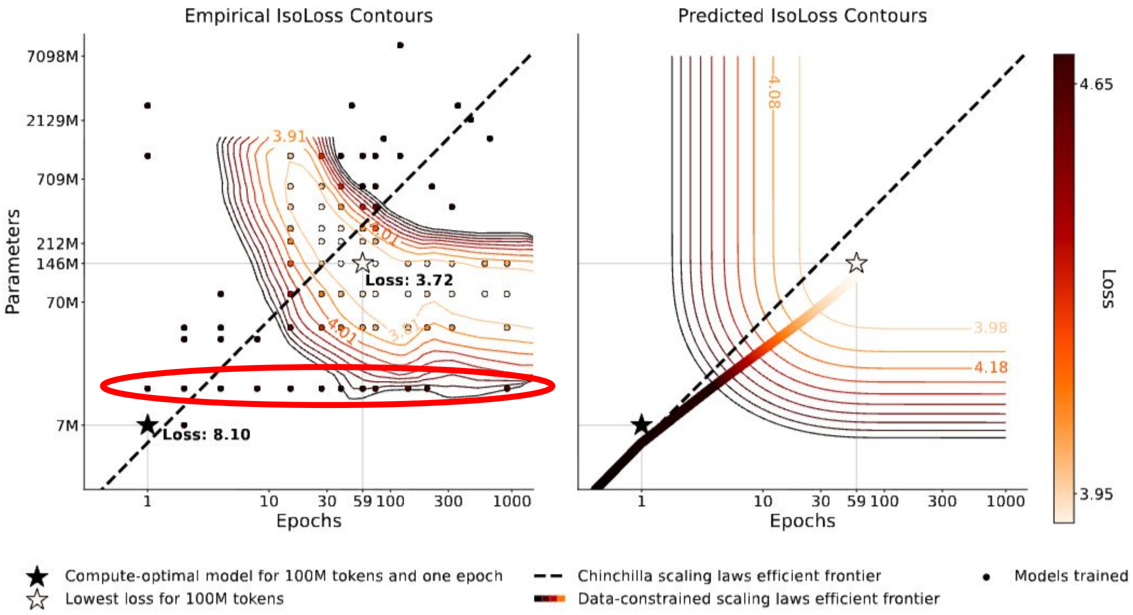


Motivation: Data might run out

Setting: Fix 100M unique tokens. If you have more compute, how do you allocate it?

https://arxiv.org/pdf/2305.16264

Scaling Laws for Data-Constrained LMs (Muennighoff et al.)

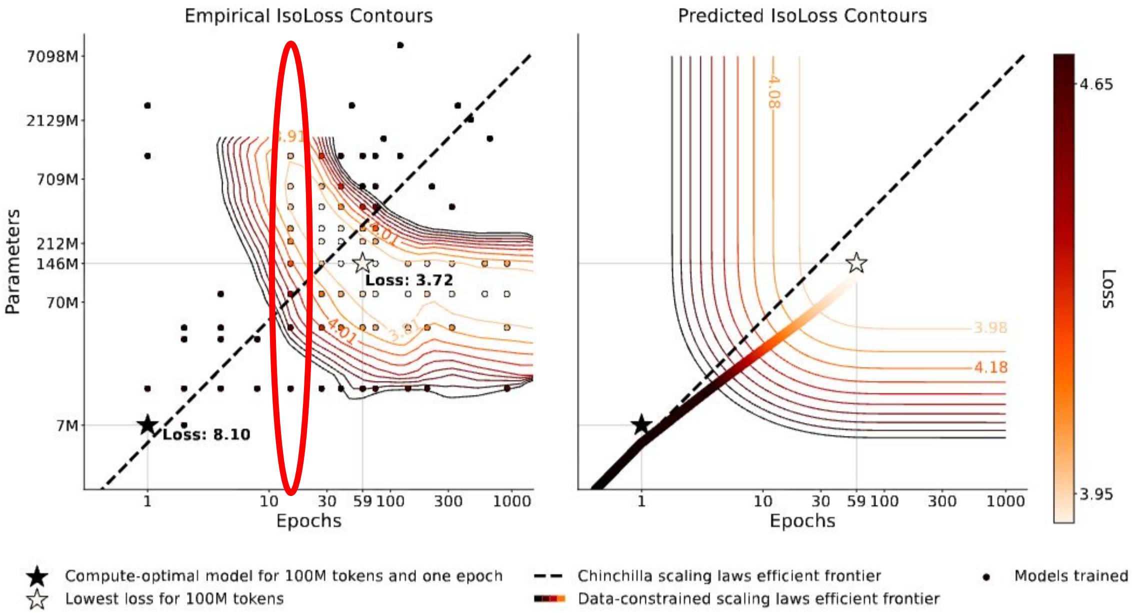


Motivation: Data might run out

Setting: Fix 100M unique tokens. If you have more compute, how do you allocate it?

https://arxiv.org/pdf/2305.16264

Scaling Laws for Data-Constrained LMs (Muennighoff et al.)

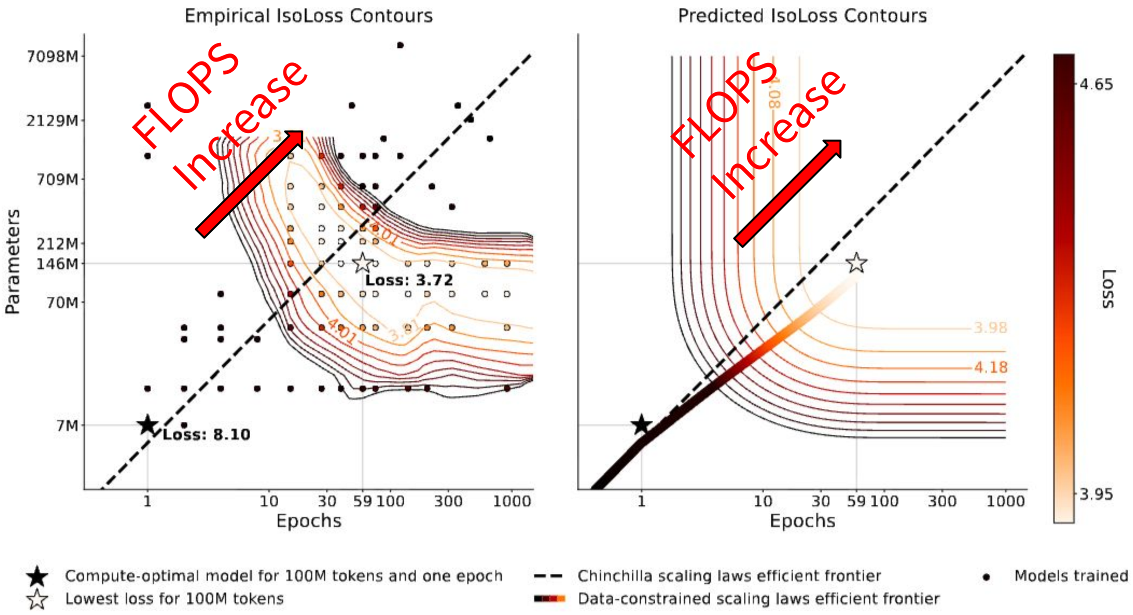


Motivation: Data might run out

Setting: Fix 100M unique tokens. If you have more compute, how do you allocate it?

https://arxiv.org/pdf/2305.16264

Scaling Laws for Data-Constrained LMs (Muennighoff et al.)



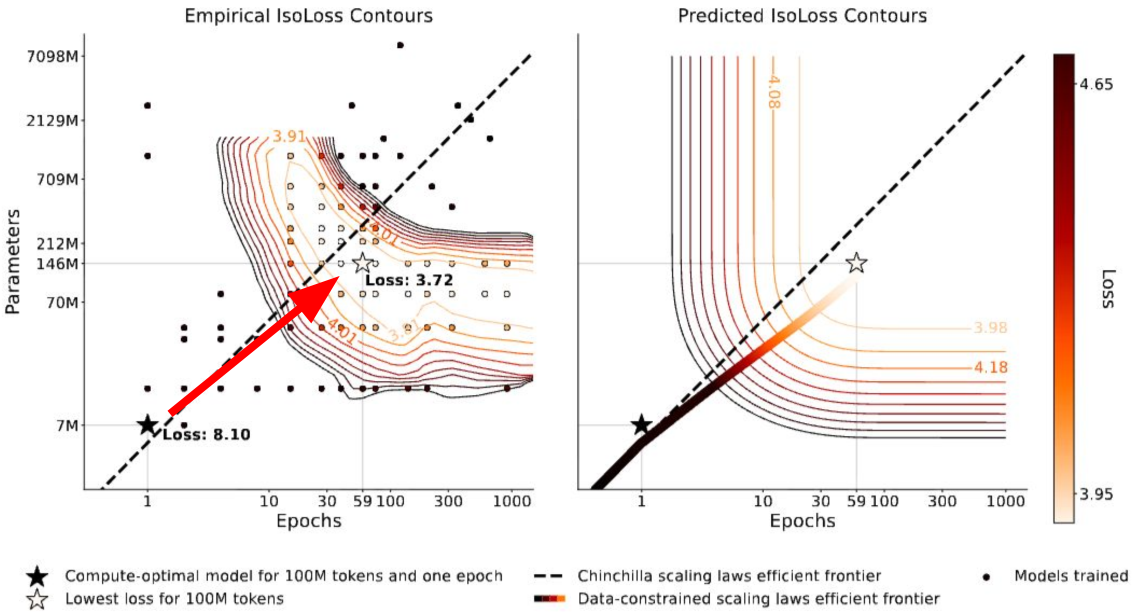
Motivation: Data might run out

Setting: Fix 100M unique tokens. If you have more compute, how do you allocate it?

Note: NOT IsoFLOP

https://arxiv.org/pdf/2305.16264

Scaling Laws for Data-Constrained LMs (Muennighoff et al.)



★ Compute-optimal model for 100M tokens and one epoch

☆ Lowest loss for 100M tokens

-- Chinchilla scaling laws efficient frontier

▬ Data-constrained scaling laws efficient frontier

● Models trained

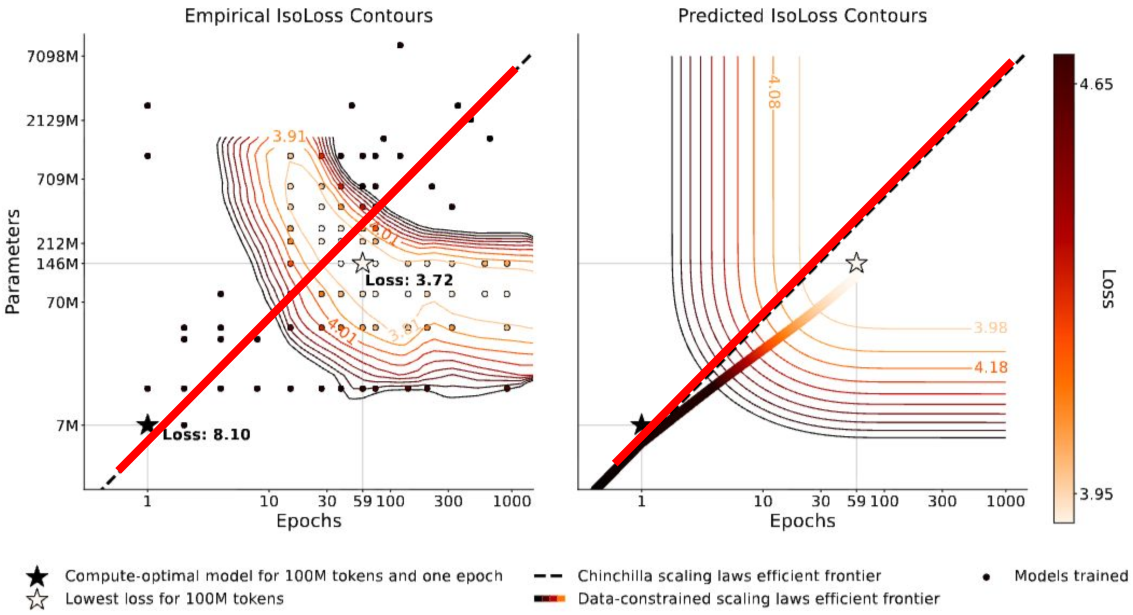
Motivation: Data might run out

Setting: Fix 100M unique tokens. If you have more compute, how do you allocate it?

Takeaway: you can squeeze more juice from data with more compute

https://arxiv.org/pdf/2305.16264

Scaling Laws for Data-Constrained LMs (Muennighoff et al.)



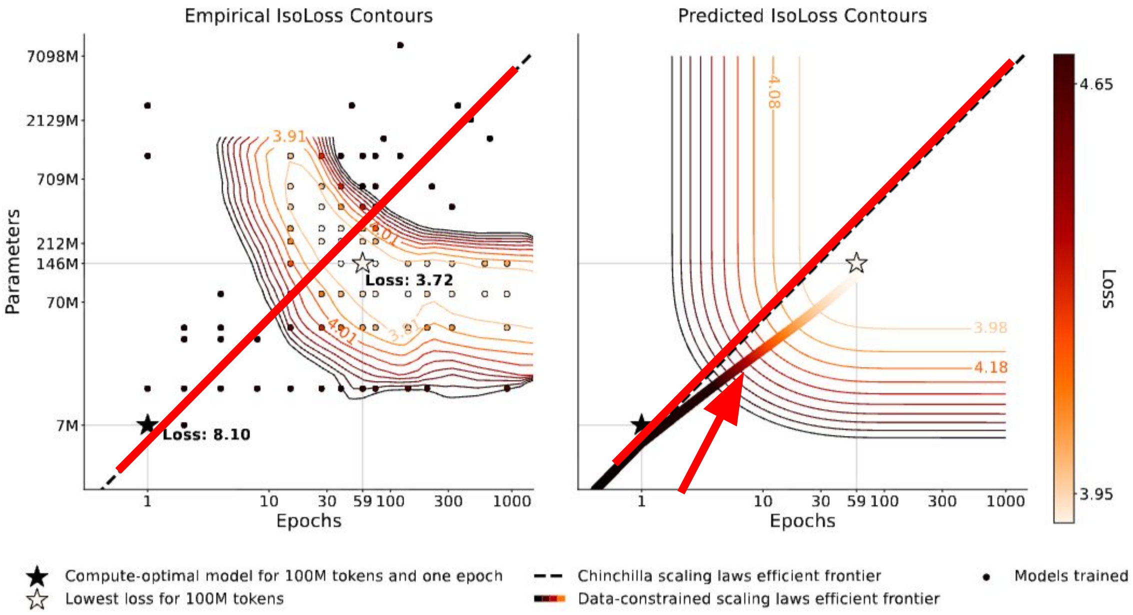
Motivation: Data might run out

Setting: Fix 100M unique tokens. If you have more compute, how do you allocate it?

Chinchilla: assumes unlimited/unique data

https://arxiv.org/pdf/2305.16264

Scaling Laws for Data-Constrained LMs (Muennighoff et al.)



Motivation: Data might run out

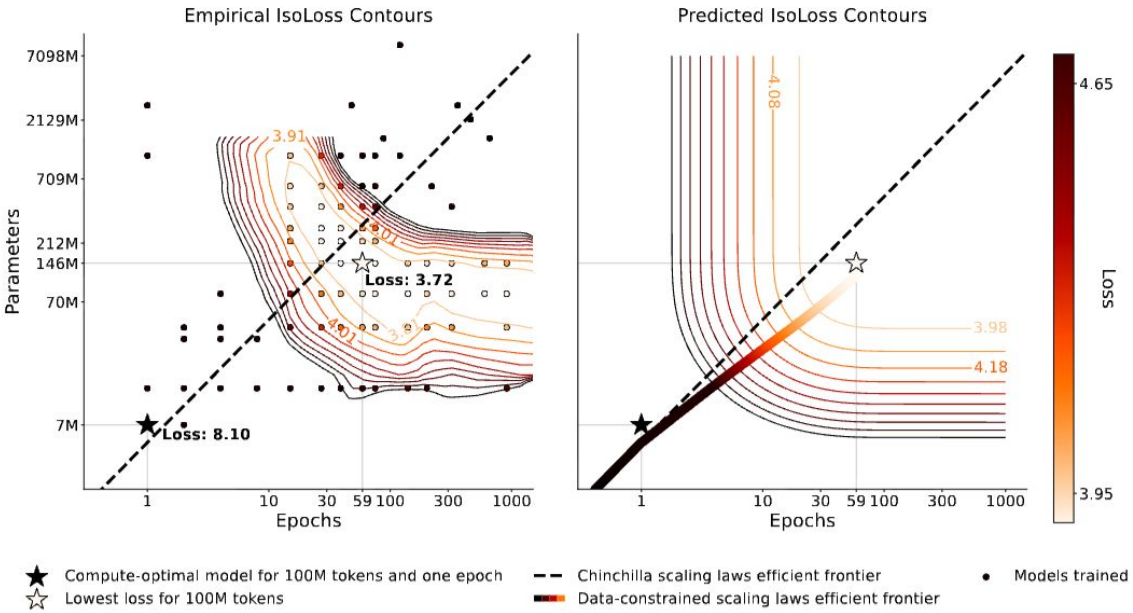
Setting: Fix 100M unique tokens. If you have more compute, how do you allocate it?

Low repetition (<4 epochs) approximates chinchilla

Excess parameters decay faster than repeated data

https://arxiv.org/pdf/2305.16264

Scaling Laws for Data-Constrained LMs (Muennighoff et al.)



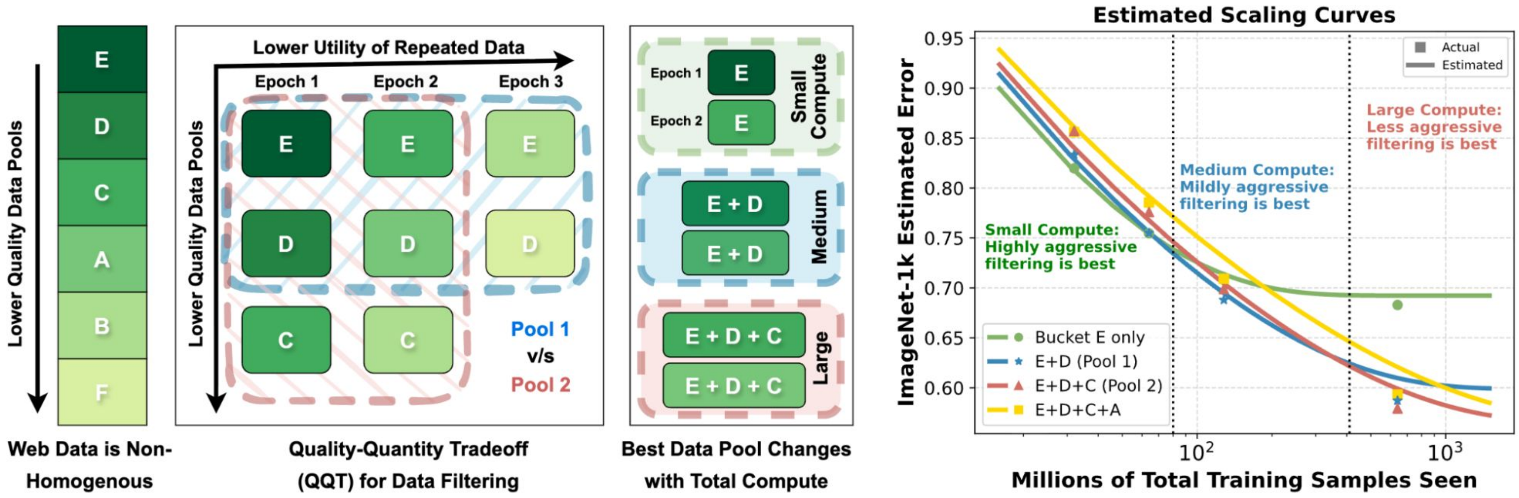
Criticisms

- Evaluates only on test loss (no down-streams)
- No considerations for data composition

https://arxiv.org/pdf/2305.16264

Scaling Laws & Data Curation (Goyal et al.)

- Chinchilla/Gadre/Muennighoff -> assumes fixed data distribution
- Goyal et al. -> data curation decisions should be developed in conjunction to the available compute



Web Data is Non-Homogenous

Quality-Quantity Tradeoff (QQT) for Data Filtering

Best Data Pool Changes with Total Compute

https://arxiv.org/pdf/2404.07177

Ongoing Research

Synthetic data

- Do classic scaling laws still hold when most data is synthetic?
- How does the way we make synthetic data (paraphrase, guided, fully-free generation) shift the scaling exponents?

Scaling Laws for Safety

- Can we reliably predict how *undesired* behaviors (toxicity, hallucination, etc) scale with parameters, repetitions, compute, etc