



CS 639: Foundation Models **Scaling I**

Fred Sala

University of Wisconsin-Madison
April 14, 2026

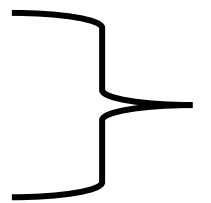


Announcements

- **Homework 4:** continue working on it!
- **Project---ongoing!**
 - Grading information here:
 - https://pages.cs.wisc.edu/~fredsala/cs639/files/project_rubric_639.pdf

• Class outline

Tuesday April 14	Scaling I
Thursday April 16	Scaling II
Tuesday April 21	Agents I
Thursday April 23	Agents II



Project Rubric

Quick run-through:

Grading: There are two items that must be turned in. First, a project proposal/pitch: a one-page pitch that describes what you plan to do. This will be worth **30 points**. Everything below will be graded for completeness:

- **[10 points]** The overarching idea you're exploring and motivation: why is this an interesting question and why are you interested in it?
- **[10 points]** Your basic hypothesis that you are evaluating/proposing,
- **[10 points]** Methodology: what techniques, tools, datasets, models, and infrastructure are required to do what you propose to do, and how you plan to tackle each of these.

Second, the final report: a report of approximately five pages (more is fine!); this will be worth **70 points**:

- **[5 points]** An introduction that explains your basic setting and problem motivation in reasonable detail. You should explain why this problem is worth studying and what we could do if it is solved.
- **[15 points]** Your hypothesis or research question (note: this can be different from the proposed one, since by now you may have advanced and made changes in your project). Your hypothesis/research question should be something that can supported (or argued against) via experiments or analysis or other means.
- **[10 points]** A reasonably detailed related work section, with citations to other papers or approaches you have explored. Explain the relationship between your project and the related work. What is similar? What is different? Why is existing work potentially insufficient to tackle your hypothesis?
- **[20 points]** A methodological (theoretical or experimental) development plan that will be used to confirm or reject your hypothesis. You will need to have detailed explanation of how you plan to tackle the research question and why the approach you are using makes sense (while other approaches would not).
- **[15 points]** Results you have obtained, along with a detailed explanation of how these confirm or reject your hypothesis (or, if that is the case, why they are neutral). **If all of your planned experiments are not complete due to time constraints, that is OK!** Explain what these planned experiments are meant to do and your expectations for what they will show once you have the opportunity to complete them. Ideally your group will have at least *some* initial results, even if limited compared to the full breadth of your work.
- **[5 points]** Conclusions and future work. What does your project say? What are good directions for continuing this work going forward?

Outline

- **Scaling Laws Intro**

- What are laws and why, regimes, idealized versions, initial findings from Kaplan et al

- **Scaling Laws Revised**

- Additional methods, new results, Chinchilla and related hypotheses

- **Beyond Scaling Laws**

- Data pruning, mixture-of-experts, etc

Outline

- **Scaling Laws Intro**

- What are laws and why, regimes, idealized versions, initial findings from Kaplan et al

- **Scaling Laws Revised**

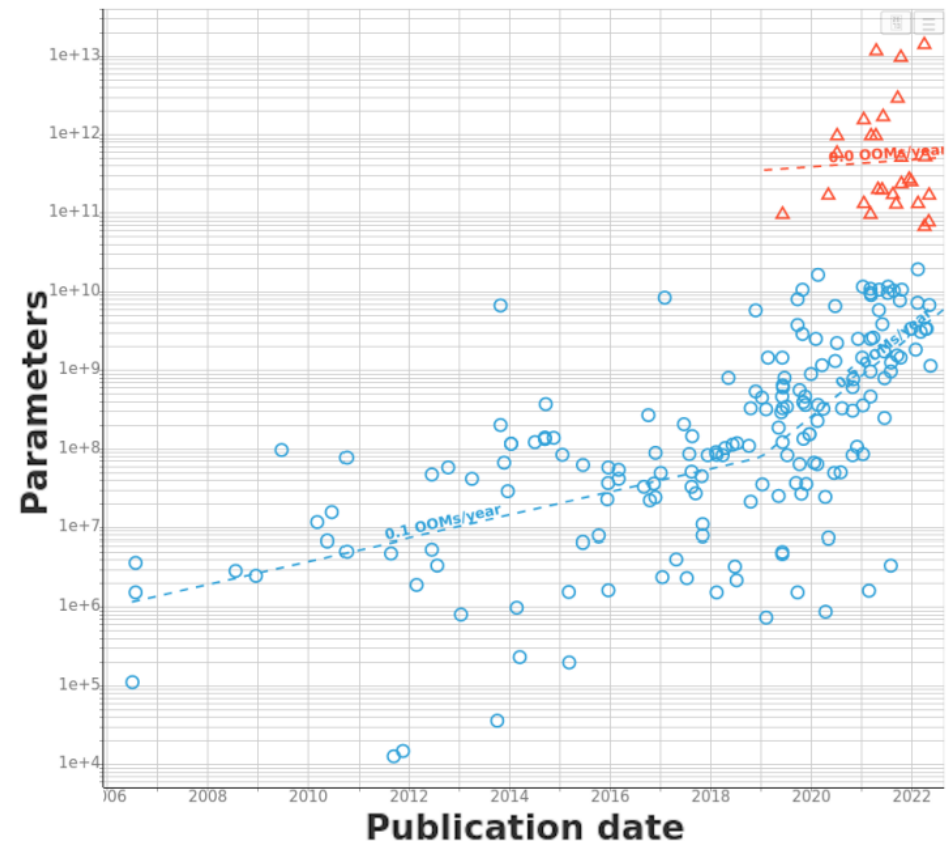
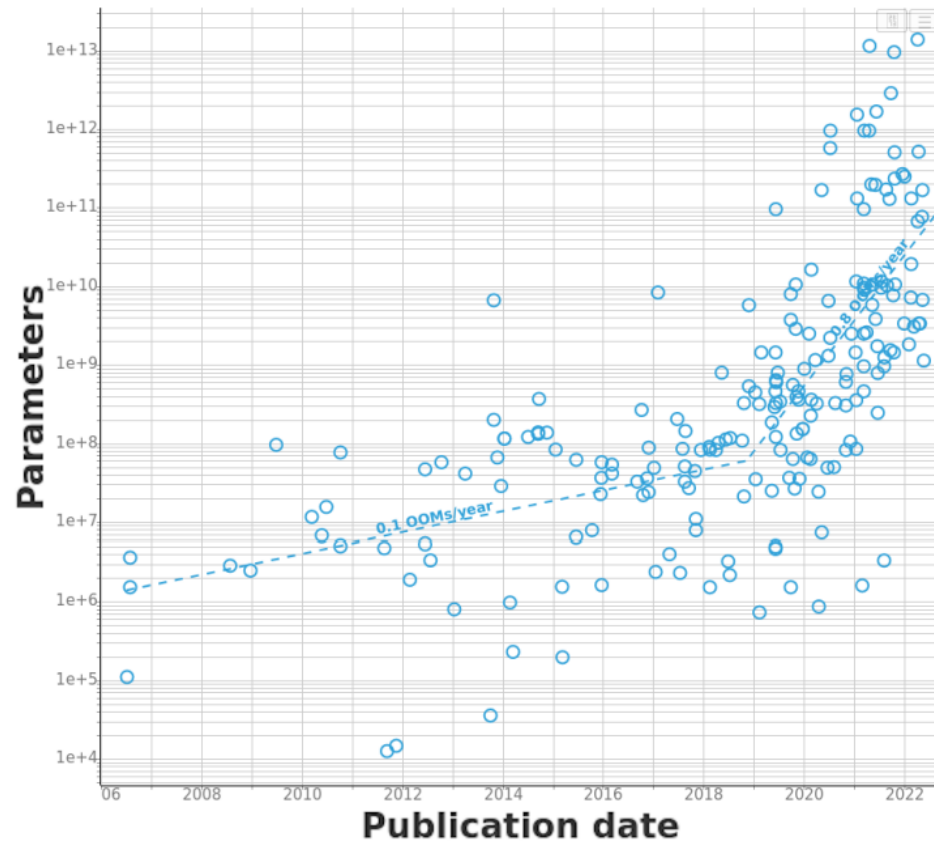
- Additional methods, new results, Chinchilla and related hypotheses

- **Beyond Scaling Laws**

- Data pruning, mixture-of-experts, etc

Trends: Models

Models have gotten bigger



Villalobos et al '22

Trends: Compute

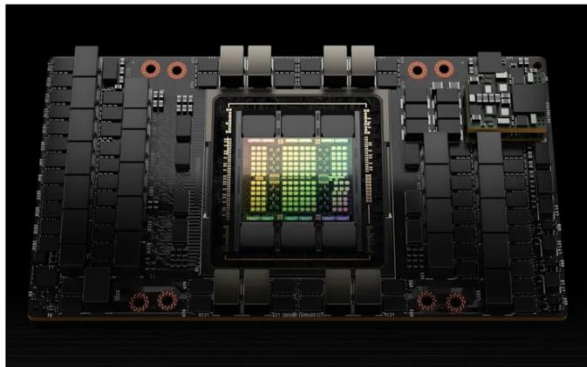
Compute has gotten bigger

Startup Builds Supercomputer with 22,000 Nvidia's H100 Compute GPUs

By Anton Shilov published July 05, 2023

The world's second highest performing supercomputer.

[f](#) [t](#) [c](#) [p](#) [r](#) [m](#) [c](#) Comments (4)



(Image credit: Nvidia)

Inflection AI, a new startup founded by the former head of DeepMind and backed

<https://www.tomshardware.com/news/startup-builds-supercomputer-with-22000-nvidias-h100-compute-gpus>

Tesla's \$300 Million AI Cluster Is Going Live Today

By Anton Shilov published August 28, 2023

Tesla is about to flip the switch on its new AI cluster, featuring 10,000 Nvidia H100 compute GPUs.

[f](#) [t](#) [c](#) [p](#) [r](#) [m](#) [c](#) Comments (23)



(Image credit: Shutterstock)

<https://www.tomshardware.com/news/teslas-dollar300-million-ai-cluster-is-going-live-today>

Home > News > Components > Graphics Cards

Zuckerberg's Meta Is Spending Billions to Buy 350,000 Nvidia H100 GPUs

In total, Meta will have the compute power equivalent to 600,000 Nvidia H100 GPUs to help it develop next-generation AI, says CEO Mark Zuckerberg.

[f](#) [t](#) [c](#) [p](#) [r](#) [m](#) [c](#) By Michael Kan January 18, 2024



(David Paul Morris/Bloomberg via Getty Images)

Mark Zuckerberg plans on acquiring 350,000 Nvidia [H100 GPUs](#) to help Meta build a next-generation AI that surpasses human-like intelligence.

<https://www.pcmag.com/news/zuckerbergs-meta-is-spending-billions-to-buy-350000-nvidia-h100-gpus>

Trends: Data

Datasets have gotten bigger

Dataset Name	Brief description	Preprocessing	Instances	Format	Default Task	Created (updated)
Statlog (Image Segmentation) Dataset	The instances were drawn randomly from a database of 7 outdoor images and hand-segmented to create a classification for every pixel.	Many features calculated.	2310	Text	Classification	1990
Caltech 101	Pictures of objects.	Detailed object outlines marked.	9146	Images	Classification, object recognition.	2003
LabelMe	Annotated pictures of scenes.	Objects outlined.	187,240	Images, text	Classification, object detection	2005
Caltech-256	Large dataset of images for object classification.	Images categorized and hand-sorted.	30,607	Images, Text	Classification, object detection	2007
ImageNet	Labeled object image database, used in the ImageNet Large Scale Visual Recognition Challenge	Labeled objects, bounding boxes, descriptive words, SIFT features	14,197,122	Images, text	Object recognition, scene recognition	2009 (2014)

wiki

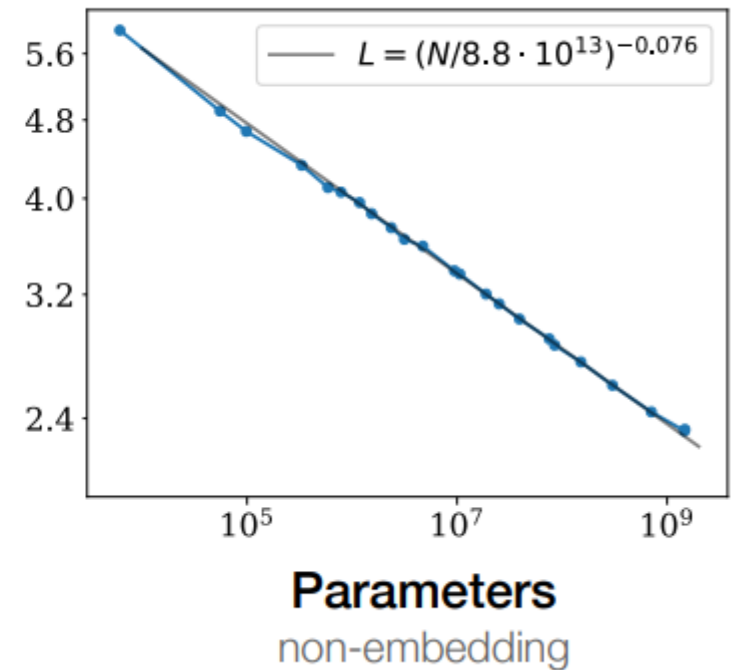
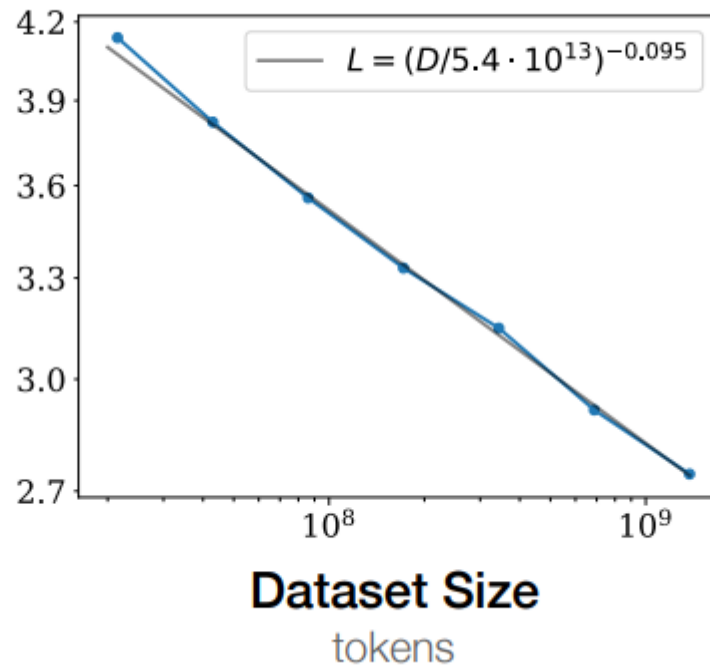
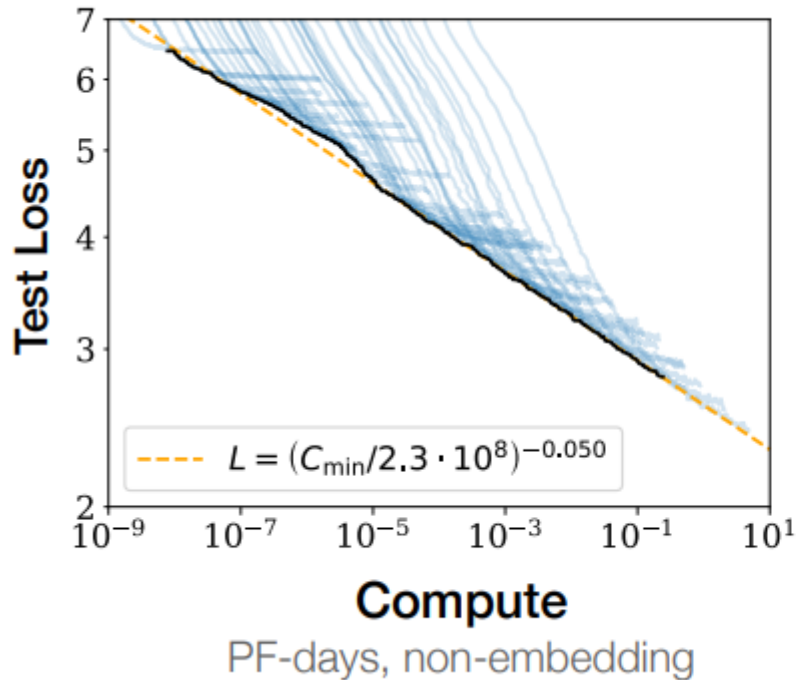
Model	Stock of data (#words)	Growth rate
Recorded speech	1.46e17 [3.41e16; 4.28e17]	5.2% [4.95%; 5.2%]
Internet users	2.01e15 [6.47e14; 6.28e15]	8.14% [7.89%; 8.14%]
Popular platforms	4.41e14 [1.21e14; 1.46e15]	8.14% [7.89%; 8.14%]
CommonCrawl	9.62e13 [4.45e13; 2.84e14]	16.68% [16.41%; 16.68%]
Indexed websites	2.21e14 [5.16e13; 6.53e15]	NA
Aggregated model	7.41e14 [6.85e13; 7.13e16]	7.15% [6.41%; 17.49%]

Villalobos et al, “Will we run out of data? An analysis of the limits of scaling datasets in Machine Learning”

Scaling Laws

We want to understand

- How performance scales with these quantities...
- And how they **interact!**

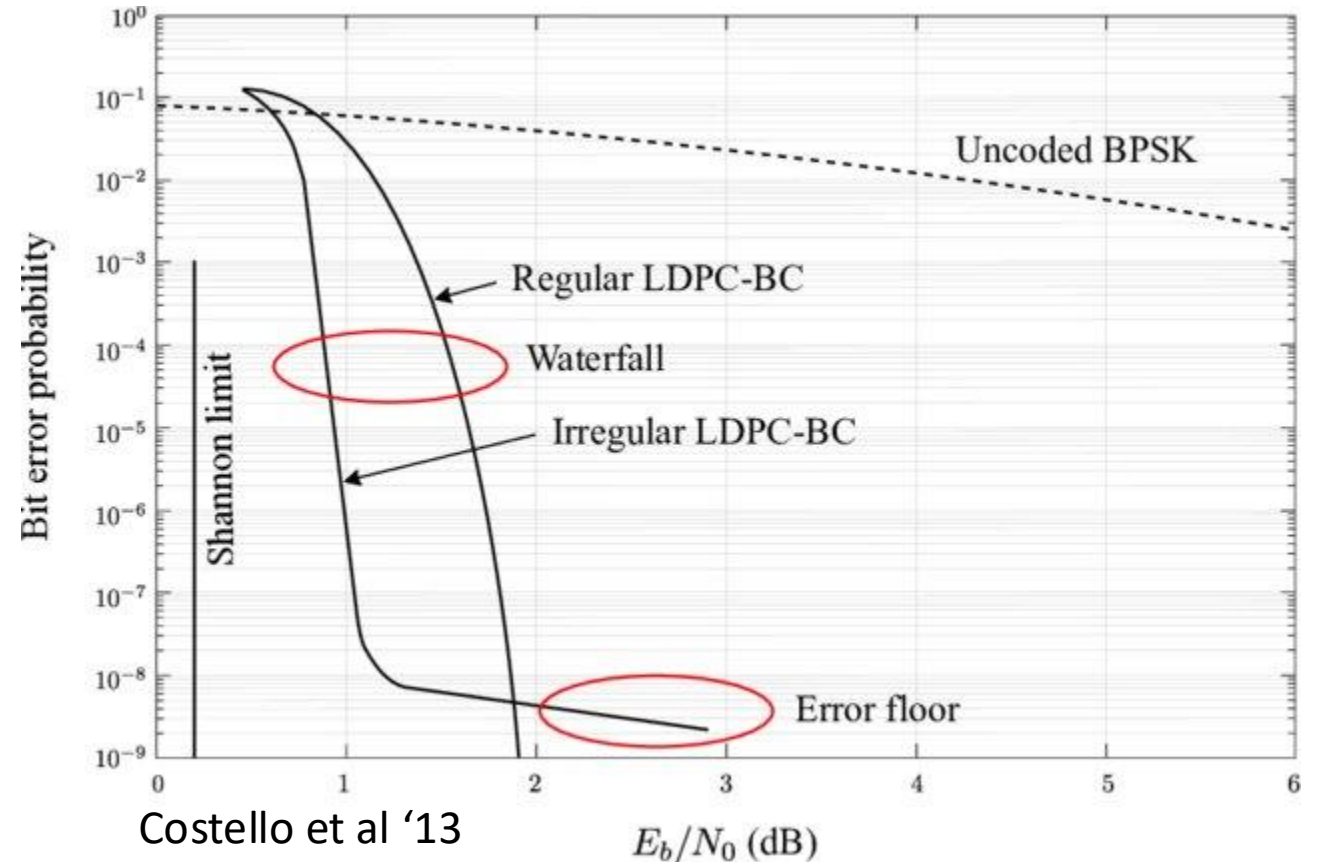


Scaling Laws

Not unique to machine learning models.

- **Note:** often have multiple “regimes”
- **Example:** LDPC and other codes

“Waterfall” regime,
“Error floor” regime



Scaling: Setup

Kaplan et al '20

Measurement units:

- **Compute:** FLOPs
- **Model size:** parameters
- **Data:** tokens

- Ranges:
- **Model size :** 768 to 1.5B (non-embedding) parameters
- **Data:** 22M to 23B tokens

Scaling Laws for Neural Language Models

Jared Kaplan *

Johns Hopkins University, OpenAI

jaredk@jhu.edu

Sam McCandlish*

OpenAI

sam@openai.com

Tom Henighan

OpenAI

henighan@openai.com

Tom B. Brown

OpenAI

tom@openai.com

Benjamin Chess

OpenAI

bchess@openai.com

Rewon Child

OpenAI

rewon@openai.com

Scott Gray

OpenAI

scott@openai.com

Alec Radford

OpenAI

alec@openai.com

Jeffrey Wu

OpenAI

jeffwu@openai.com

Dario Amodei

OpenAI

damodei@openai.com

Compute: FLOPS

FLOPs: a measure of computing performance

- “floating point operations per second”
- Our neural network operations involve adding and multiplying real numbers → flops
 - Note: standard approach 32 bit floating point
 - **Popular area of research:** smaller precision or mixed precision training, inference, or both

September 2022	\$0.02	\$0.02	RTX 4090	Nvidia's RTX 4090 is listed as having a peak performance of 82.6 TFLOPS (1.32 PFLOPS at 8-bit precision) at a retail price of \$1599. ^[87]
May 2023	\$0.01	\$0.01	Radeon RX 7600	AMD's RX 7600 is listed as having a peak performance of 21.5 TFLOPS at a retail price of \$269. ^[88]

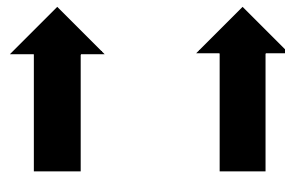
Wiki

Scaling: Power Laws

How to model relationships measured?

- Power laws

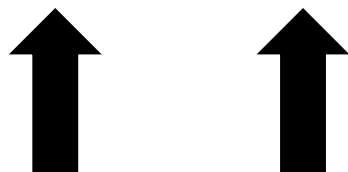
$$f(x) = ax^{-k}$$



Coefficient **Exponent**

- In our case, for model size and training to convergence,

$$L(N) = (N_c/N)^{\alpha_N}; \quad \alpha_N \sim 0.076, \quad N_c \sim 8.8 \times 10^{13}$$

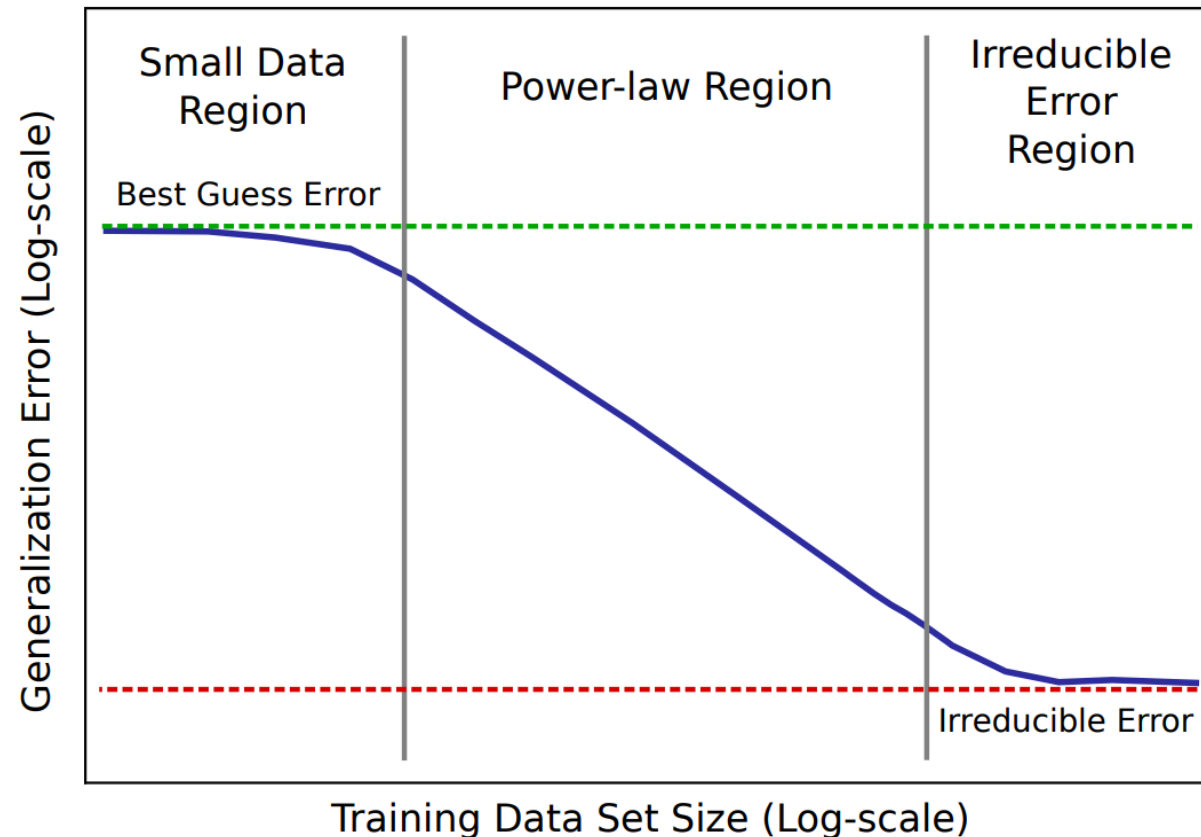


Coefficient **Exponent**

Scaling: Power Laws

Not a new idea. For data: hypothetical power-law like scaling

- **Note:** different regimes



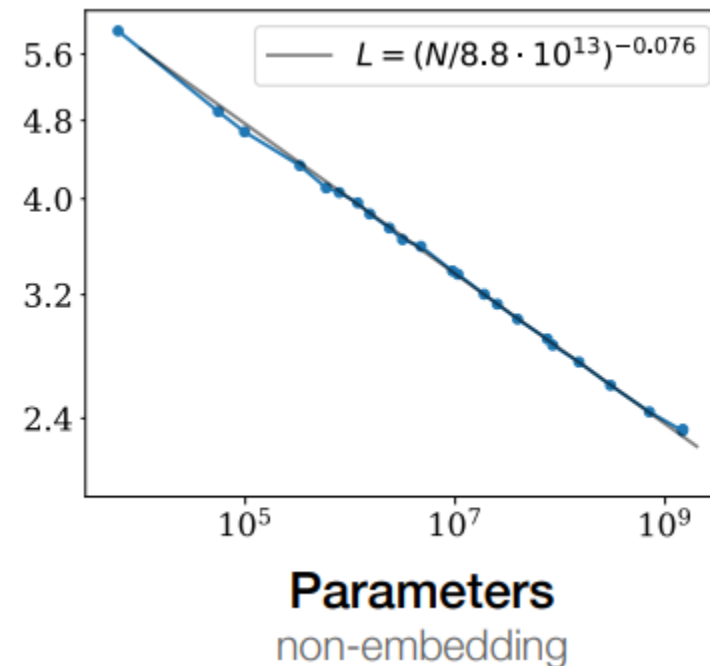
Scaling: Varying the Model Size

Let's see this in detail.

Kaplan et al '20. Fix the dataset (large).

- **Vary model size: 769 to 1.5B**
- Measure test loss
- Fit the curve as before:

$$L(N) = (N_c/N)^{\alpha_N} ; \quad \alpha_N \sim 0.076, \quad N_c \sim 8.8 \times 10^{13}$$

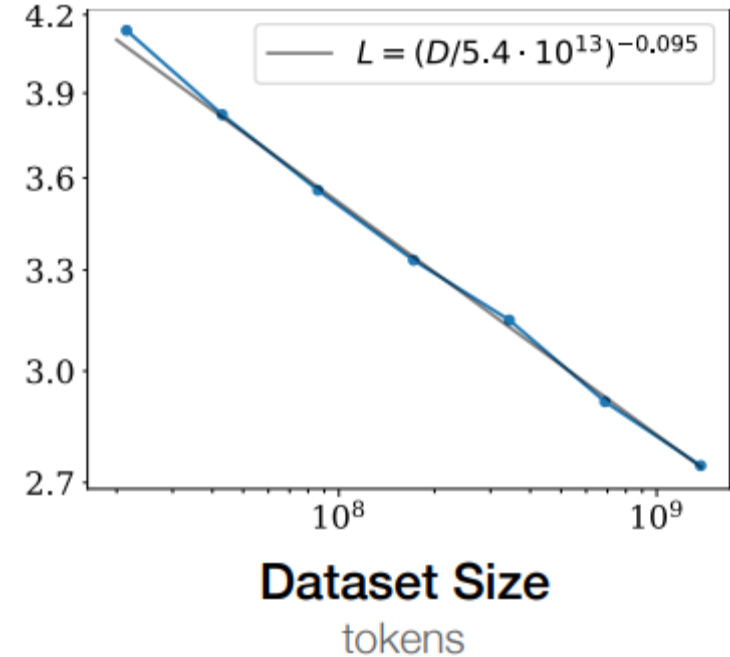


Scaling: Varying the Dataset

Same idea, but for data.

Fix the model size (large).

- **Vary Data:** 22M to 23B tokens
- Measure test loss
- Again fit a curve



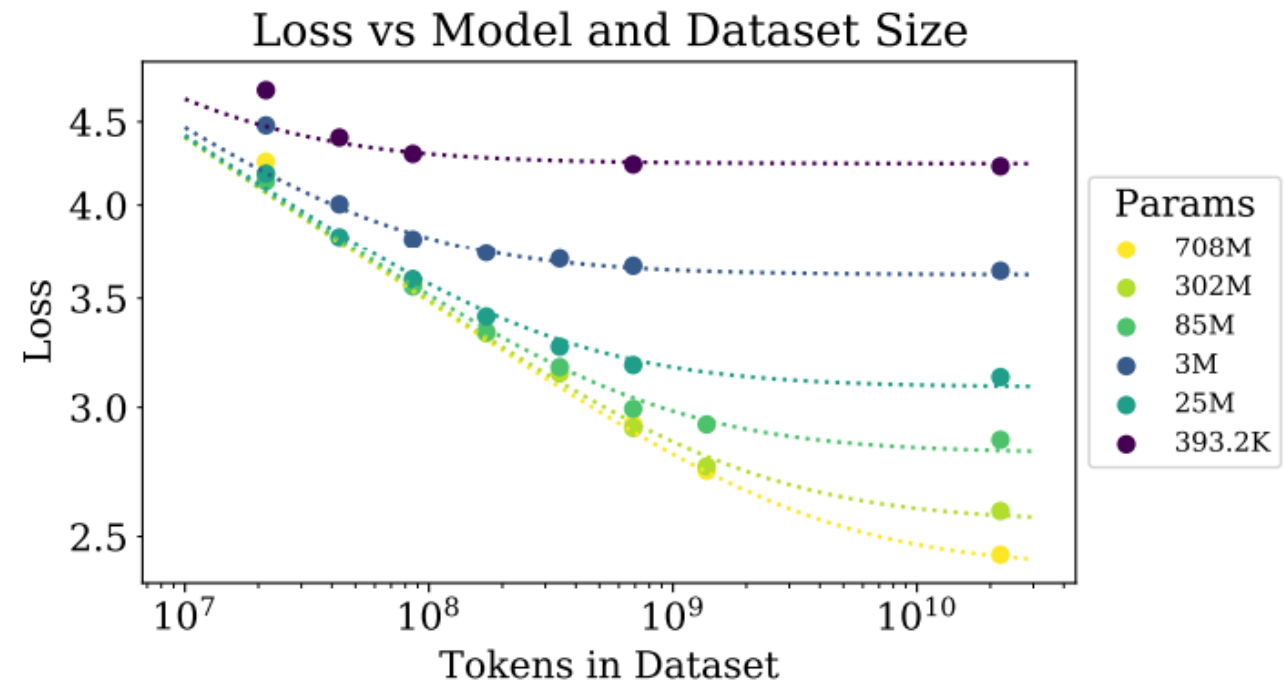
$$L(D) = (D_c/D)^{\alpha_D}; \quad \alpha_D \sim 0.095, \quad D_c \sim 5.4 \times 10^{13} \text{ (tokens)}$$

Scaling: Interactions

What about the effect of both model size and data?

- **Why?** Need to figure out what to prioritize: get more data or increase the model size?
 - “as we increase the model size, we should increase the dataset size sublinearly according to $D \propto N^{\alpha_N/\alpha_D} \sim N^{0.74}$ ”

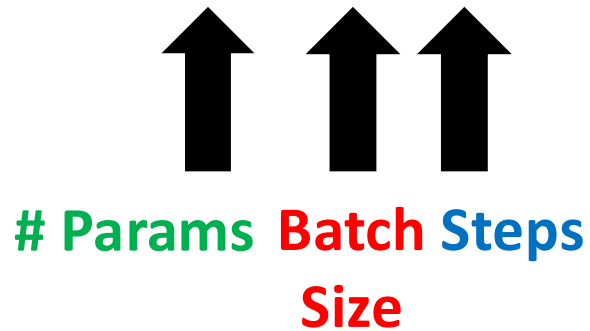
$$L(N, D) = \left[\left(\frac{N_c}{N} \right)^{\frac{\alpha_N}{\alpha_D}} + \frac{D_c}{D} \right]^{\alpha_D}$$



Scaling: Compute

How much compute do we need?

- **Note:** not independent of the data/model size!
- Rough equation: $C = 6 N \times B \times S$



- C is a direct function of model size.
 - Batch size varies (existing heuristics for optimal batch size).
 - Steps depend on stopping rules

Scaling: Compute

What are the interactions?

- Using the **critical batch size** (optimizes the speed/efficiency tradeoff).

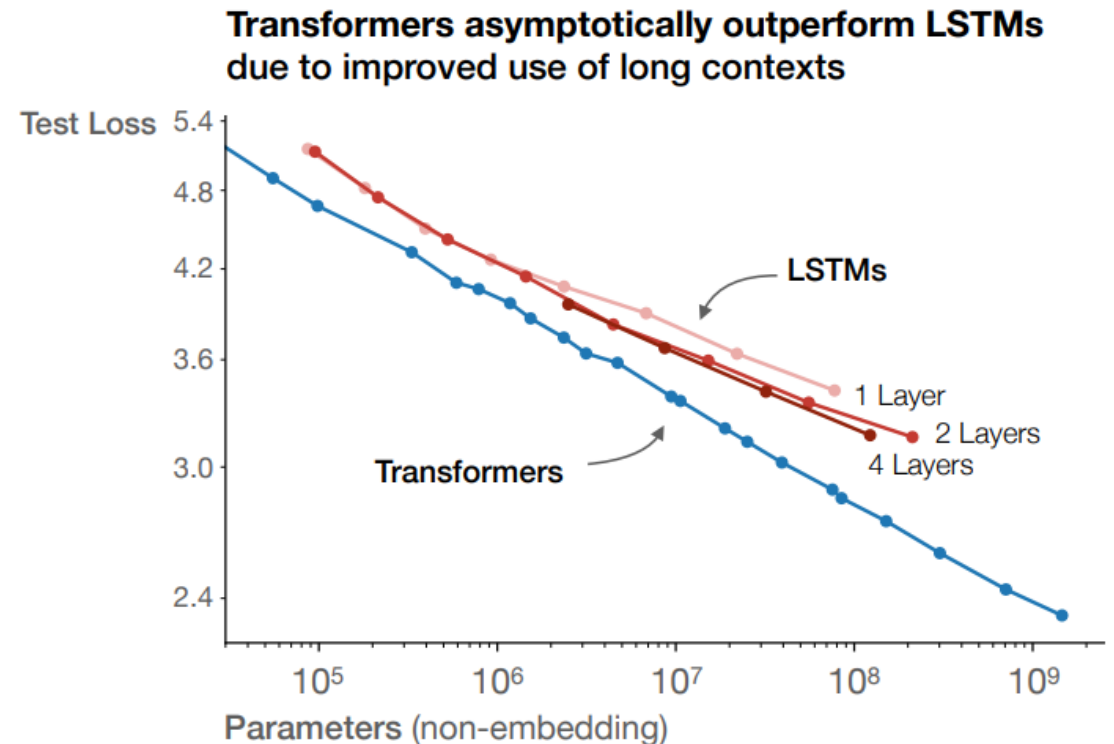
$$N \propto C^{\alpha_C^{\min}} / \alpha_N, \quad B \propto C^{\alpha_C^{\min}} / \alpha_B, \quad S \propto C^{\alpha_C^{\min}} / \alpha_S, \quad D = B \cdot S$$

- Empirically optimal results: $N \propto C^{0.73}$, $B \propto C^{0.24}$, and $S \propto C^{0.03}$
- “As the computational budget C increases, it should be spent primarily on larger models, without dramatic increases in training time or dataset size”

Scaling: Architectures

What about choosing various architectures?

- Compare **transformers** vs **LSTMs**
- Change parameter counts, #layers
 - Fixed dataset (WebText2)
- Transformers win here
 - Some recent work challenges this



Scaling: Predicting

All of this requires huge numbers of training runs...

- But, if the laws are reliable, can:
- Train smaller models,
- Obtain a scaling law,
- Make design decisions based on this law.





Break & Questions

Outline

- **Scaling Laws Intro**

- What are laws and why, regimes, idealized versions, initial findings from Kaplan et al

- **Scaling Laws Revised**

- Additional methods, new results, Chinchilla and related hypotheses

- **Beyond Scaling Laws**

- Data pruning, mixture-of-experts, etc

Scaling: How Universal Is This?

Kaplan et al made certain choices,

- Results used early stopping, etc.
- One particular learning rate schedule
- Scaling law results may change with different choices!
- Hoffman et al '22: another exploration with **different results**.



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*

*Equal contributions

We investigate the optimal model size and number of tokens for training a transformer language model under a given compute budget. We find that current large language models are significantly under-trained, a consequence of the recent focus on scaling language models whilst keeping the amount of training data constant. By training over 400 language models ranging from 70 million to over 16 billion parameters on 5 to 500 billion tokens, we find that for compute-optimal training, the model size and the number of training tokens should be scaled equally: for every doubling of model size the number of training tokens should also be doubled. We test this hypothesis by training a predicted compute-optimal model, *Chinchilla*, that uses the same compute budget as *Gopher* but with 70B parameters and 4× more data. *Chinchilla* uniformly and significantly outperforms *Gopher* (280B), GPT-3 (175B), Jurassic-1 (178B), and Megatron-Turing NLG (530B) on a large range of downstream evaluation tasks. This also means that *Chinchilla* uses substantially less compute for fine-tuning and inference, greatly facilitating downstream usage. As a highlight, *Chinchilla* reaches a state-of-the-art average accuracy of 67.5% on the MMLU benchmark, greater than a 7% improvement over *Gopher*.

SL2: Approach #1: Minimum Over Curves

For each number of parameters (range: 70M to 10B),

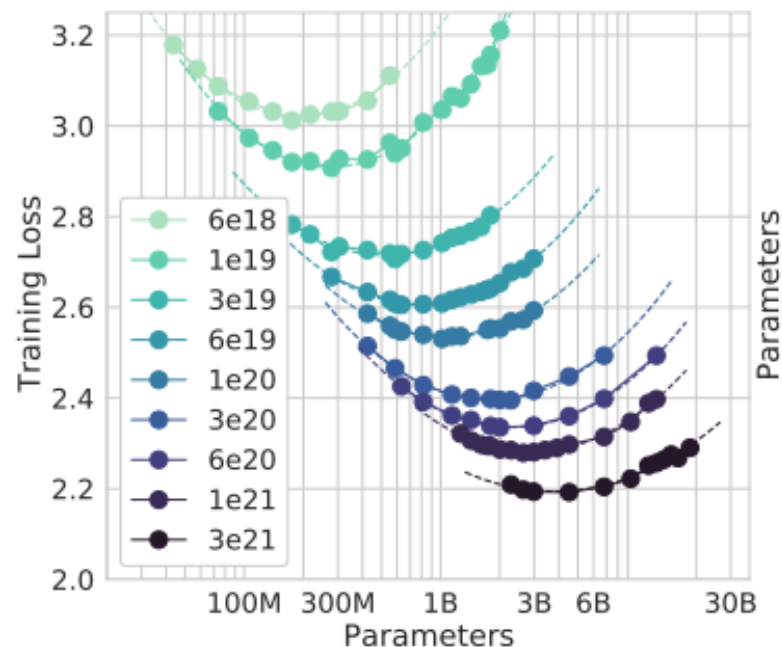
- Vary # of training steps,
- 4 training sequences, take overall minimum
- **Results:**

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)
Kaplan et al. (2020)	0.73	0.27

SL2: Approach #2: IsoFLOP Profiles

Vary model size for a fixed set of FLOP counts

- Obtain best performance for fixed FLOP at various models, use to obtain curve



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)
Kaplan et al. (2020)	0.73	0.27

SL2: Approach #3: Direct Fitting

Fit the function (inspired by classical risk bounds)

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

Results:

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

SL2 Conclusion

Note all results fairly similar:

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

“All three approaches suggest that as compute budget increases, model size and the amount of training data should be increased in approximately equal proportions”

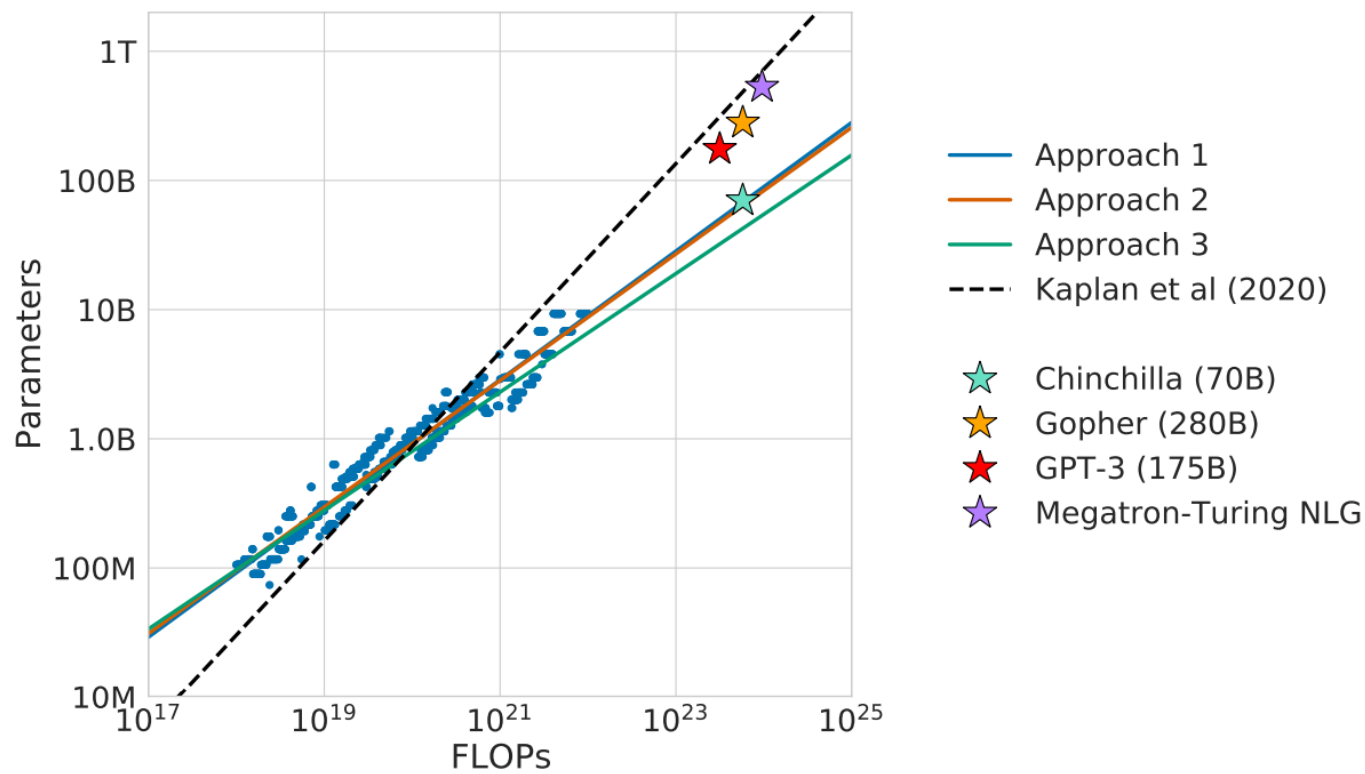
- Quite different from Kaplan et al!

SL2 Chinchilla

What are the implications?

- For a particular (large) compute budget, very massive models are not the way to go,
- “**Smaller**” is better.
- Chinchilla model: 70B parameters, 1.4T tokens
 - Comparison against Gopher: same compute in FLOPs, but much larger

Random	25.0%
Average human rater	34.5%
GPT-3 5-shot	43.9%
<i>Gopher</i> 5-shot	60.0%
<i>Chinchilla</i> 5-shot	67.6%
Average human expert performance	89.8%



Reconciling Differences & Practical Use

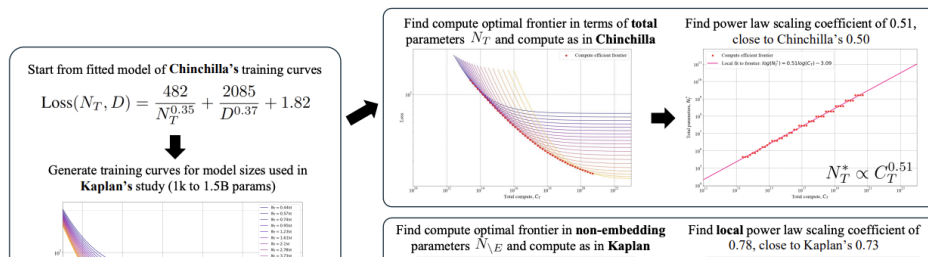
Reconciling Kaplan and Chinchilla Scaling Laws

Tim Pearce *Microsoft Research*

Jinyeop Song *MIT*

Abstract

Kaplan et al. (2020) ('Kaplan') and Hoffmann et al. (2022) ('Chinchilla') studied the scaling behavior of transformers trained on next-token language prediction. These studies produced different estimates for how the number of parameters (N) and training tokens (D) should be set to achieve the lowest possible loss for a given compute budget (C). Kaplan: $N_{\text{optimal}} \propto C^{0.73}$, Chinchilla: $N_{\text{optimal}} \propto C^{0.50}$. This paper finds that much of this discrepancy can be attributed to Kaplan counting non-embedding rather than total parameters, combined with their analysis being performed at small scale. Simulating the Chinchilla study under these conditions produces biased scaling coefficients close to Kaplan's. Hence, this paper reaffirms Chinchilla's scaling coefficients, by explaining the primary cause of Kaplan's original overestimation. As a second contribution, the paper explains differences in the reported relationships between loss and compute. These findings lead us to recommend that future scaling studies use total parameters and compute. ¹



Reproducing some scaling laws results from Chinchilla. Can't get the numbers to match exactly, but can still be used as a rough guide to help determine compute-optimal models. Also contains related utilities for calculating flops and param counts.

```
[1]: import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
%matplotlib inline
```

params

First some parameter calculations:

```
[2]: def gpt_params(seq_len, vocab_size, d_model, num_heads, num_layers):
      """ Given GPT config calculate total number of parameters """
      ffw_size = 4*d_model # in GPT the number of intermediate features is always 4*d_model
      # token and position embeddings
      embeddings = d_model * vocab_size + d_model * seq_len
      # transformer blocks
      attention = 3*d_model**2 + 3*d_model # weights and biases
      attproj = d_model**2 + d_model
      ffw = d_model*(ffw_size) + ffw_size
      ffwproj = ffw_size*d_model + d_model
      layernorms = 2*2*d_model
      # dense
      ln_f = 2*d_model
      dense = d_model*vocab_size # note: no bias here
      # note: embeddings are not included in the param count!
      total_params = num_layers*(attention + attproj + ffw + ffwproj + layernorms) + ln_f + dense
      return total_params

      gpt2 = dict(seq_len = 1024, vocab_size = 50257, d_model = 768, num_heads = 12, num_layers = 12)
      gpt_params(**gpt2)/1e6
```

[2]: 123.653376

OpenAI reports gpt2 (small) as having 124M params, so this is a match. Also, loading the OpenAI weights into nanoGPT and then calling `model.parameters()` exactly matches the above number and verifies the implementation. Now Chinchilla parameters:

https://github.com/karpathy/nanoGPT/blob/master/scaling_laws.ipynb



Break & Questions

Outline

- **Scaling Laws Intro**

- What are laws and why, regimes, idealized versions, initial findings from Kaplan et al

- **Scaling Laws Revised**

- Additional methods, new results, Chinchilla and related hypotheses

- **Beyond Scaling Laws**

- Data pruning and others

Back to Universality

Even if we could estimate these law parameters correctly, are we stuck with the implications?

- Maybe not!

- Better **data** via pruning

**Beyond neural scaling laws:
beating power law scaling via data pruning**

Ben Sorscher^{*1}

Robert Geirhos^{*2}

Shashank Shekhar³

Surya Ganguli^{1,3§}

Ari S. Morcos^{3§}

^{*}equal contribution

¹Department of Applied Physics, Stanford University

²University of Tübingen

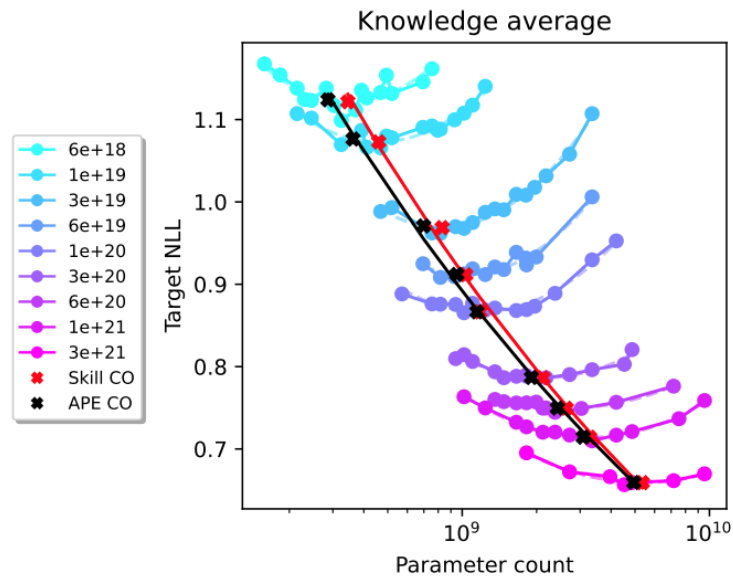
³Meta AI (FAIR)

[§]Joint senior authors

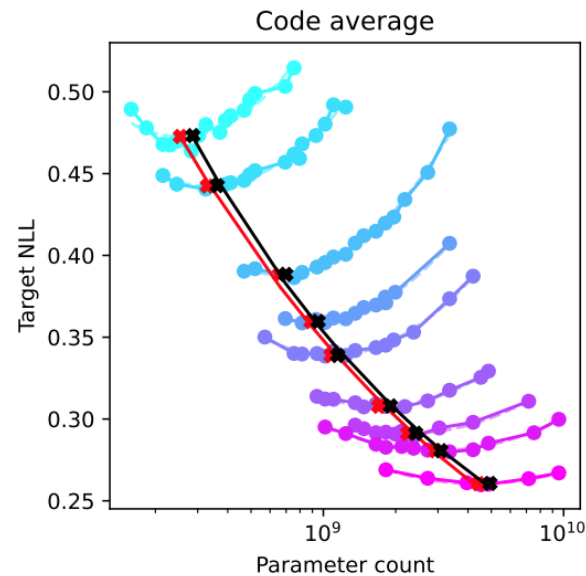
Beyond Single Data Types

Do all data/tasks show the same scaling?

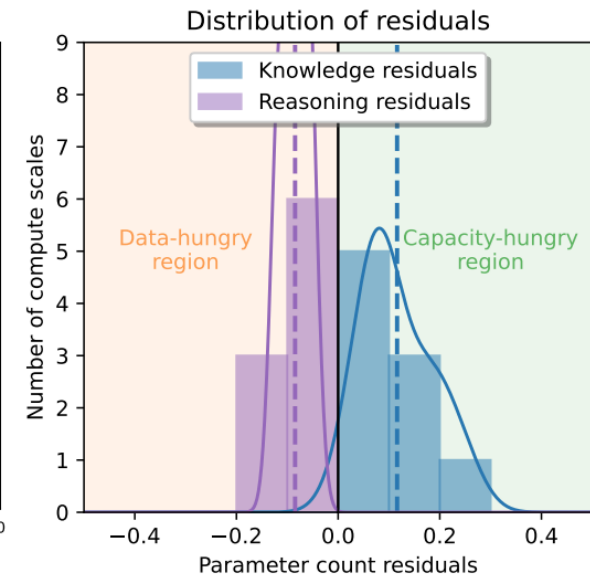
- Maybe not!
- Reasoning versus knowledge:
 - Knowledge: more **data-hungry**, reasoning: more **compute-hungry**



(a) Knowledge QA.



(b) Code.

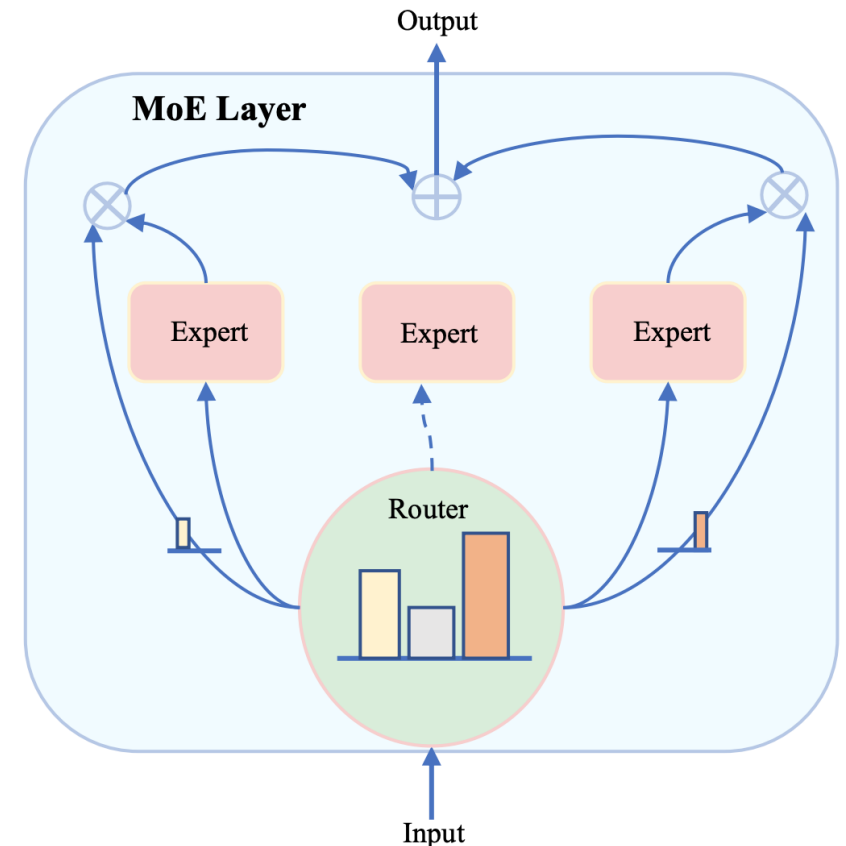


(c) Data- vs capacity-hunger.

Beyond Standard Architectures: MoE

Another approach: modify architecture to decouple scaling from inference-time costs

- A popular approach: mixture of experts (MoE)
- Router points to path to take
- **Active** parameters much smaller than **full model-size**: can scale up models further





Thank You!