CSE 599J:

Scaling Laws

Hamish Ivison and Hannah Lin

UNIVERSITY of WASHINGTON

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, Laurent Sifre

DeepMind

Advent of LLMs

		# parameters	# tokens
2020	**GPT-3**	175B	300B
2021	**Jurassic-1**	178B	300B
	Gopher	280B	300B
2022	**Megatron-Turing NLG**	530B	270B
	LaMDA	137B	168B

Table data from [1, 2, 3, 4, 5]

Problem: Resource Costs

- Training LLMs comes with a high compute and energy cost
- Cost increases with model size

Model	Hardware	Power (W)	Hours	kWh·PUE	CO_2e	Cloud compute cost
$\text{Transformer}_{base}$	P100x8	1415.78	12	27	26	\$41–\$140
Transformer_{big}	P100x8	1515.43	84	201	192	\$289–\$981
ELMo	P100x3	517.66	336	275	262	\$433–\$1472
BERT_{base}	V100x64	12,041.51	79	1507	1438	\$3751–\$12,571
BERT_{base}	TPUv2x16	—	96	—	—	\$2074–\$6912
NAS	P100x8	1515.43	274,120	656,347	626,155	\$942,973–\$3,201,722
NAS	TPUv2x1	—	32,623	—	—	\$44,055–\$146,848
GPT-2	TPUv3x32	—	168	—	—	\$12,902–\$43,008

Table 3 from Energy and Policy Considerations for Deep Learning in NLP [6]

Problem: Resource Costs

- FLOPs: floating point operations

$$\mathbf{C \approx 6ND}$$

  - *C*: non-embedding training compute ←***this is constrained***
  - *N*: number of model parameters
  - *D*: number of tokens

- Goal: **maximize model performance** by finding optimal values for **N** and **D**

The Question

C

- “Given a fixed FLOPs budget, how should one trade-off model size and the number of training tokens?”

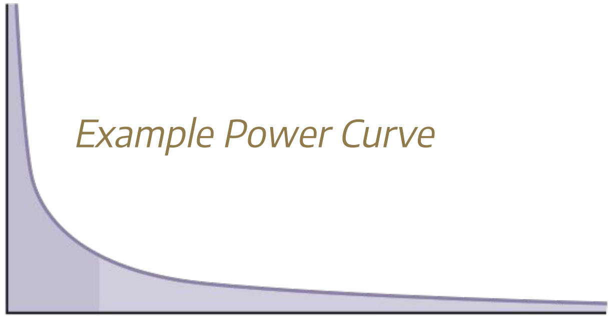
N

D

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

Power Law (Before Chinchilla)

- Scaling Laws for Neural Language Models, Kaplan et al. [7]
- Power-law relationship between training test loss and:
  - ***C***: non-embedding training compute
  - ***N***: number of model parameters
  - ***D***: number of tokens



Example Power Curve

Power Law (Before Chinchilla)

- Large models should not be trained to their lowest possible loss

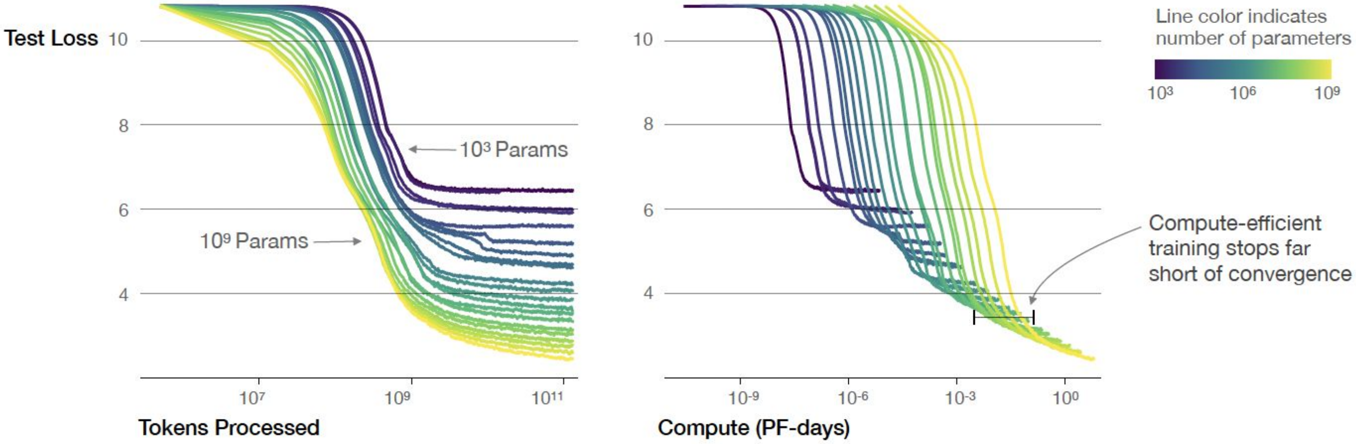


Figure 2 from Scaling Laws for Neural Language Models[7]

Power Law (Before Chinchilla)

- If doubling ***N*** with a fixed batch size, increase ***D*** by 1.7x
- If doubling ***N*** with a compute-efficient batch size, increase ***D*** by 1.3x

In other words:

$$N \propto C^{0.73} \quad \text{and} \quad D \propto C^{0.27}$$

Key Contribution

> $\boldsymbol{N}$ and $\boldsymbol{D}$ should scale **equally**

$$N \propto C^{\cancel{0.73}\ \sim 0.50} \quad \text{and} \quad D \propto C^{\cancel{0.27}\ \sim 0.50}$$

→ many past models can be reduced in size

Deep Dive: 3-Pronged Approach

> Approach 1: fix *N*, vary *D*

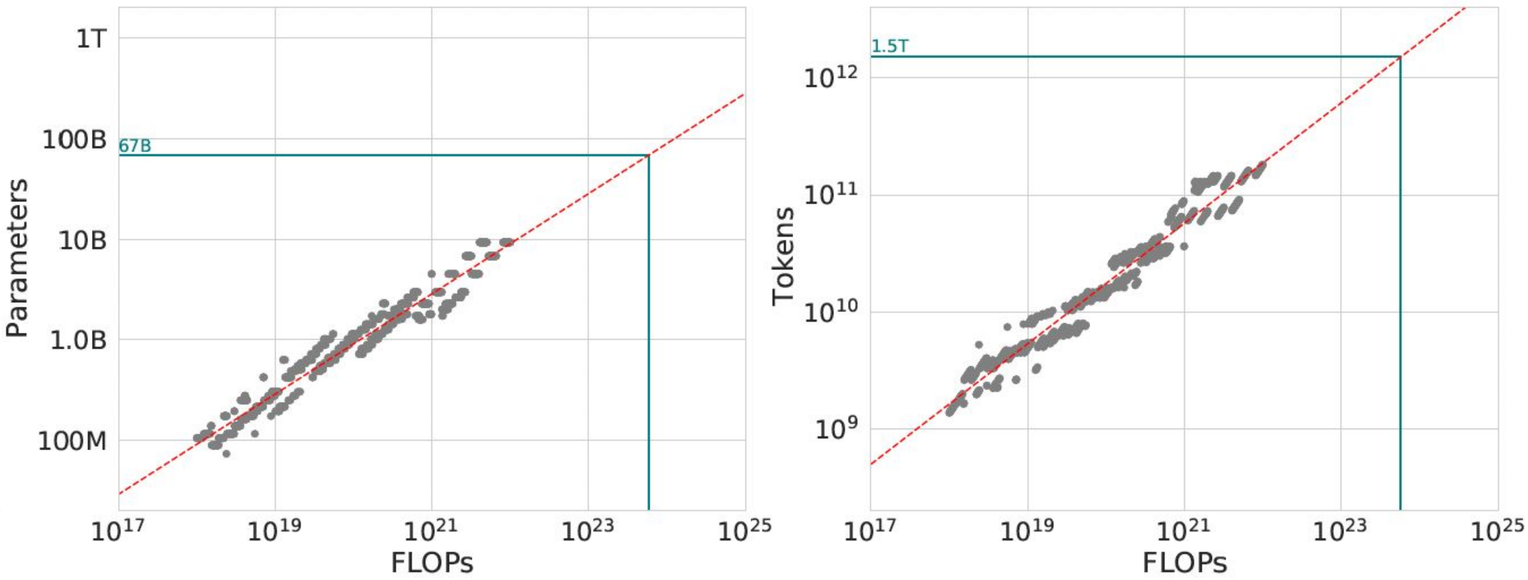


Figure 2. Training curve envelope.

Deep Dive: 3-Pronged Approach

> Approach 2: IsoFLOP profiles
> (Fix *C*, vary *N* and *D*)

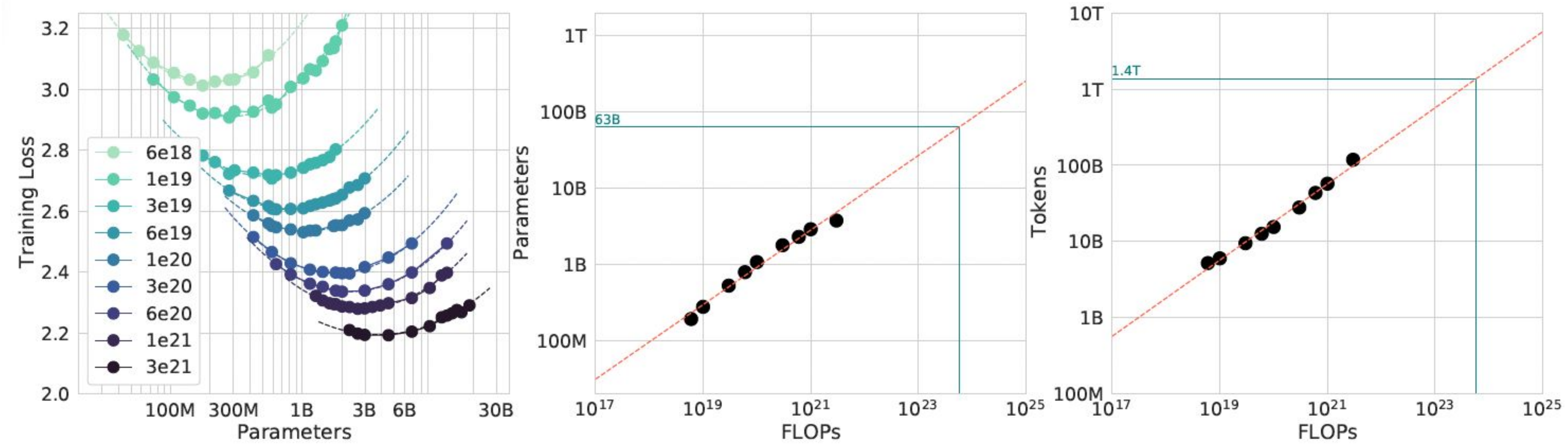


Figure 3. IsoFLOP Curves.

Deep Dive: 3-Pronged Approach

- Approach 3: Fit a parametric loss function
  - Estimate parameters using the optimization algorithm L-BFGS

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

Deep Dive: 3-Pronged Approach

$N \propto C^a$ and $D \propto C^b$

	a	b
Kaplan et al.	0.73	0.27
Approach 1	0.50	0.50
Approach 2	0.49	0.51
Approach 3	0.46	0.54

Deep Dive: 3-Pronged Approach

"current large language models are significantly undertrained"

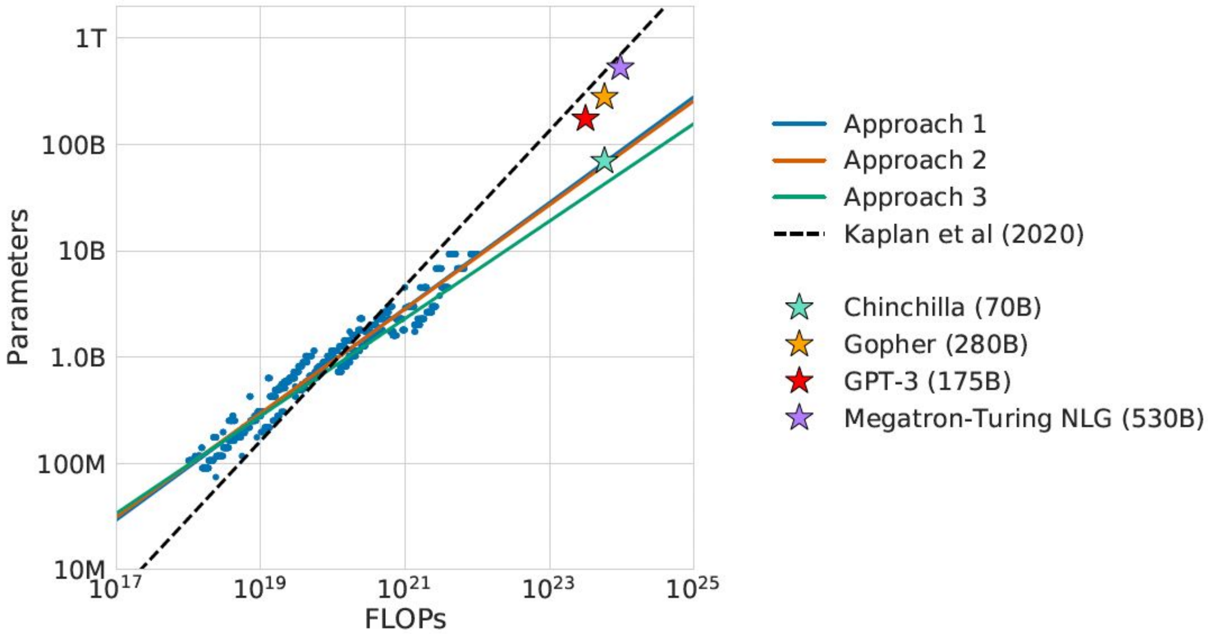


Figure 1. Models with scaling law predictions. [8]

Before Chinchilla: Gopher

- 280B parameters
- 300B training tokens

Chinchilla

- Decreases *N* by 4x
  and increases *D* by 4x
- ~~280B~~ 70B parameters
- ~~300B~~ 1.4T training tokens

Performance: Chinchilla vs. Gopher

- Chinchilla performs better across the board, including on downstream tasks

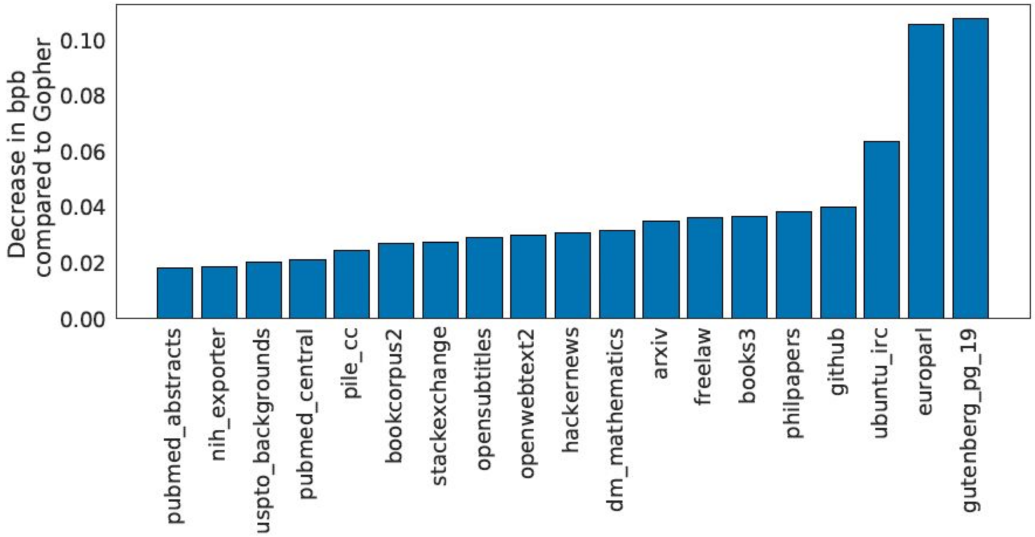


Figure 5. Pile Evaluation with bits-per-byte improvement. [8]

Performance: Chinchilla vs. Gopher

- Less affected by bias and toxicity than Gopher

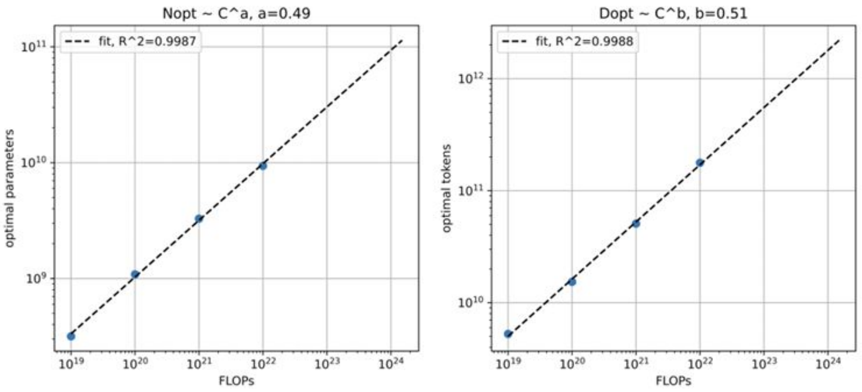
	Chinchilla	*Gopher*
All	78.3%	71.4%
Male	71.2%	68.0%
Female	79.6%	71.3%
Neutral	84.2%	75.0%

Table 10: Winogender results showing pronounce resolution. [8]

Impact & Response

> Debates on general **applicability of scaling laws**

→ PaLM 2: *"We validate this study for larger amounts of compute and similarly find that* ***data and model size should be scaled roughly 1:1****"* [9]



PaLM Technical Report Figure 5. [9]

Impact & Response

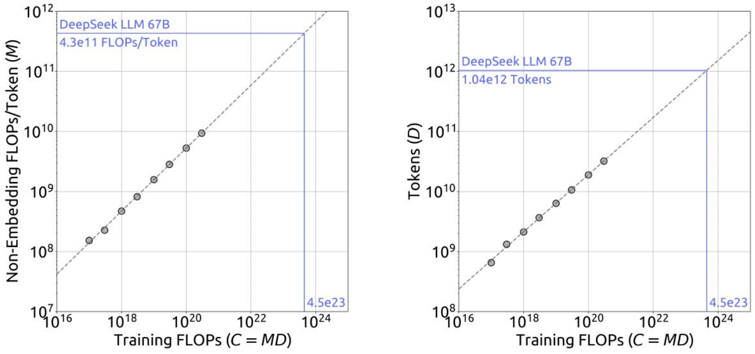
> Debates on general **applicability of scaling laws**

[1]

→ DeepSeek presents a **new scaling law**:[19]

$$M \propto C^{0.52} \quad \text{and} \quad D \propto C^{0.46}$$

(M = FLOPs/token)



DeepSeek IsoFLOP Figure 5. [19]

Impact & Response

*"This calls for … a high **focus on dataset quality**."*

- DeepSeek

  (1) confirms dataset quality matters

  (2) shows higher data quality means *more compute should be allocated to model scaling*

Approach	Coeff. a where $N_{\text{opt}}(M_{\text{opt}}) \propto C^a$	Coeff. b where $D_{\text{opt}} \propto C^b$
OpenAI (OpenWebText2)	0.73	0.27
Chinchilla (MassiveText)	0.49	0.51
Ours (Early Data)	0.450	0.550
Ours (Current Data)	0.524	0.476
Ours (OpenWebText2)	0.578	0.422

higher quality

Table 4 on scaling coefficients from DeepSeek. [19]

Impact & Response

> Brought more attention to **importance of dataset size**... along with some worries:

tokens? How much text data *is* there, exactly?

2. are we running out of data?

It is frustratingly hard to

From LessWrong [10]

Will we run out of data? An analysis of the limits of scaling datasets in Machine Learning

Pablo Villalobos*, Jaime Sevilla*†, Lennart Heim*§, Tamay Besiroglu*‡, Marius Hobbhahn *¶, Anson Ho*

Some sorts of a rebuttal. [11]

Scaling Data-Constrained Language Models

Niklas Muennighoff, Alexander M. Rush, Boaz Barak, Teven Le Scao, Aleksandra Piktus, Nouamane Tazi, Sampo Pyysalo, Thomas Wolf, Colin Raffel

Hugging Face

HARVARD UNIVERSITY

UNIVERSITY OF TURKU

What happens when we run out of tokens?

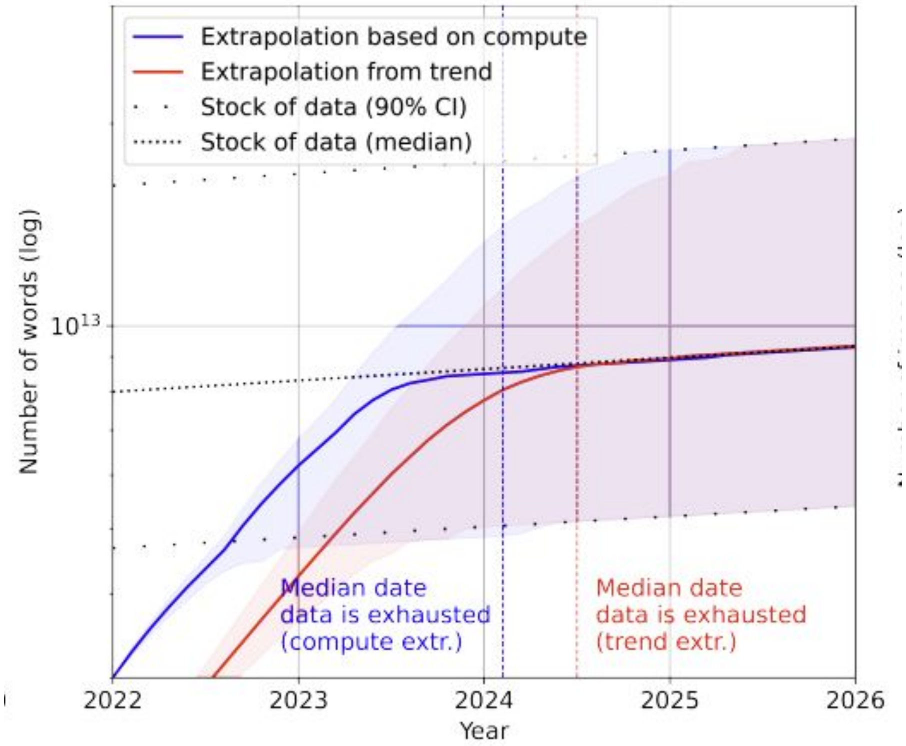


Fig. 1 (centre) from [11]

What if we are working in a data-constrained domain?

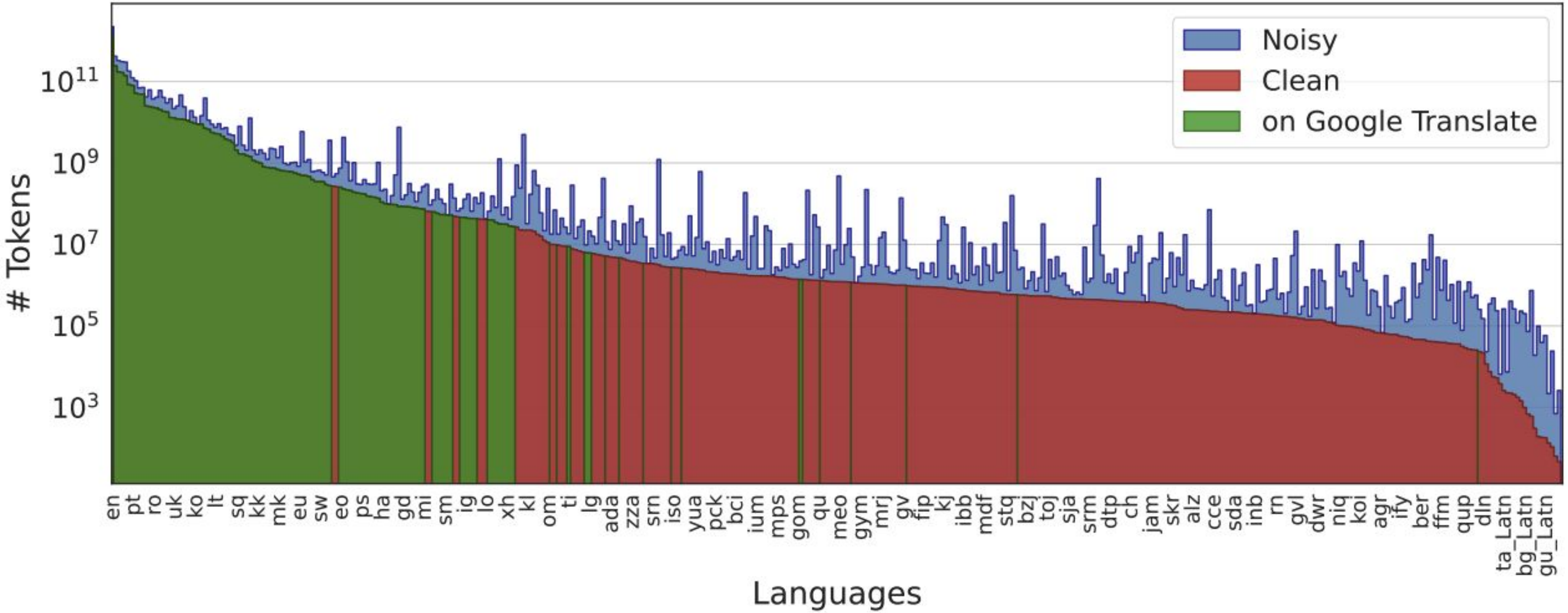


Figure 1 from [12]

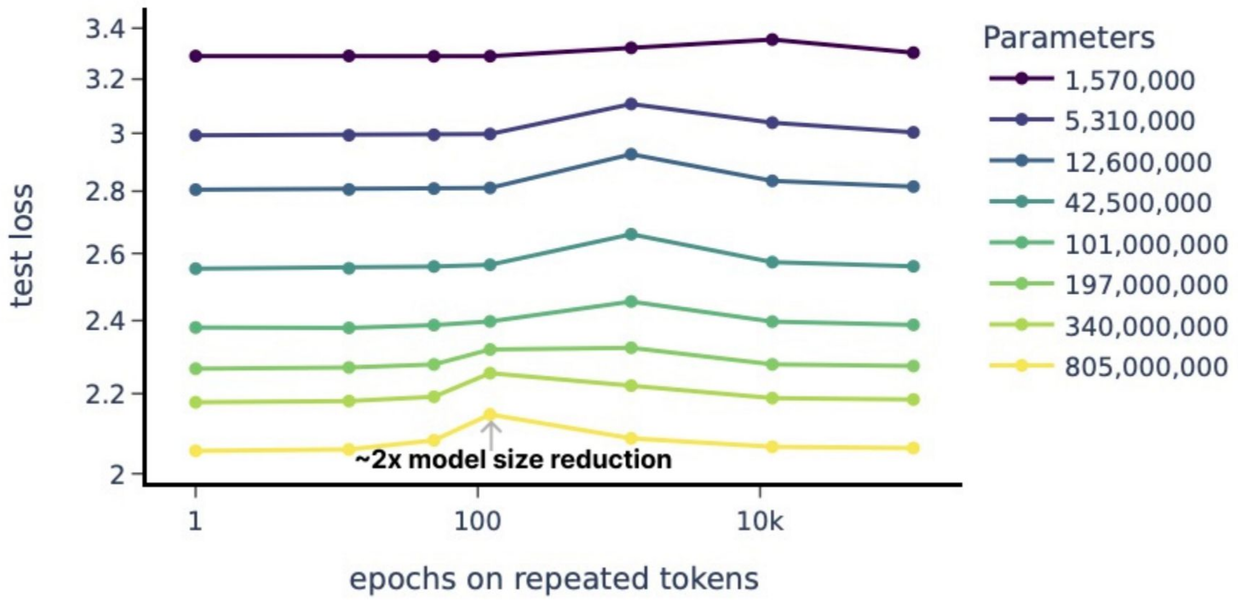
How to scale further data-constrained settings?

1. Repeat data (multiple epochs)
2. Add non-natural-language data (e.g. code)
3. Include “lower-quality” data (e.g. remove filters)

How to scale further data-constrained settings?

1. Repeat data (multiple epochs)
2. Add non-natural-language data (e.g. code)
3. Include "lower-quality" data (e.g. remove filters)

Repeated data considered harmful



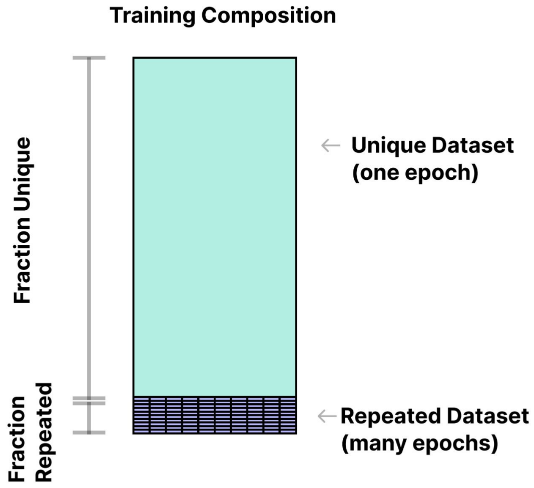
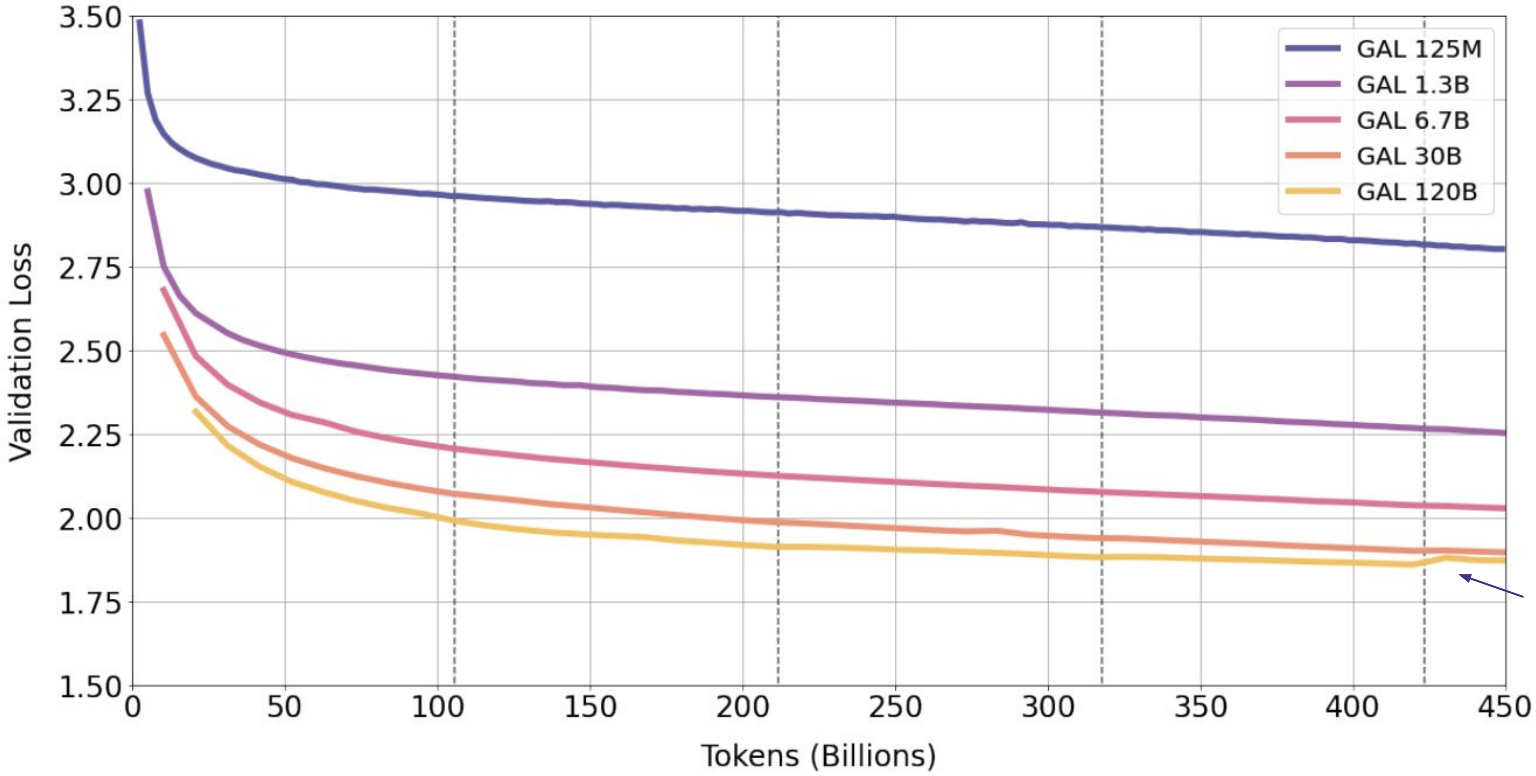


Figure 2 (left) from [18]

But is repeated data actually bad?



Validation loss does not jump until 5th epoch!

Figure 6 from [13]

Given a fixed FLOPs budget, how should one trade-off model size and the number of training tokens?

Given a fixed FLOPs budget **and fixed amount of distinct data**, how should one trade-off model size and **the number of epochs**?

Strategy: Parametric model

Take chinchilla model and augment for repeated data!

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

Modelling repeated data

Data utility as **exponential decay:**

$$D' = U + (1-\delta)U + (1-\delta)^2U + \cdots + (1-\delta)^{R_D}U$$

"effective" tokens

unique tokens

decay rate

number of repetitions

"effective tokens" = Repeating *U* tokens R_D times is roughly the same as having *D'* unique tokens.

Modelling repeated data

Data utility as **exponential decay:**

$$D' = U + (1-\delta)U + (1-\delta)^2 U + \cdots + (1-\delta)^{R_D} U$$

“effective” tokens

unique tokens

decay rate

number of repetitions

Using sum of a geometric series:

$$D' = U + (1-\delta)U\,\frac{(1-(1-\delta)^{R_D})}{\delta}$$

We could directly estimate δ, but the paper goes further to define it in terms of “optimal number of repetitions”

Modeling repeated data

Define R_D^* ~= 'maximum useful number of repeats'

$$
\begin{aligned}
D' &= U + (1-\delta)U\frac{(1-(1-\delta)^{R_D})}{\delta} \\
&= U + U \cdot R_D^* \cdot (1 - e^{-R_D/R_D^*})
\end{aligned}
$$

R* = 0 → repeated data is useless

R* = ∞ → repeated data is as good as new data

Modelling repeated parameters

We perform the same steps for ‘repeating parameters’ to model how excess parameters behave, yielding a similar equation:

actual number of repetitions

$$\left(U_N + U_N R_N^* \left(1 - e^{\frac{-R_N}{R_N^*}}\right)\right)$$

unique params ~= “optimal” parameters

optimal repetitions for params

Modelling repeated data and parameters

Recall chinchilla "law":

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

Now we have a way to represent 'effective' parameters and data, we replace N, U with those:

$$L(U_N, U_D, R_N, R_D) = \frac{A}{(U_N + U_N R_N^*(1 - e^{\frac{-R_N}{R_N^*}}))^{\alpha}} + \frac{B}{(U_D + U_D R_D^*(1 - e^{\frac{-R_D}{R_D^*}}))^{\beta}} + E \quad (14)$$

unique params/ data

repeated params/ data

Fitting our model

We set *A*, *B*, *α*, *β*, U_N using the chinchilla law, then learn the optimal repetitions for parameters and data:

$$L(U_N, U_D, R_N, R_D) = \frac{A}{(U_N + U_N R_N^*(1 - e^{\frac{-R_N}{R_N^*}}))^\alpha} + \frac{B}{(U_D + U_D R_D^*(1 - e^{\frac{-R_D}{R_D^*}}))^\beta} + E \tag{14}$$

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

$$\text{where } U_N = U_D \cdot 0.051 \tag{17}$$

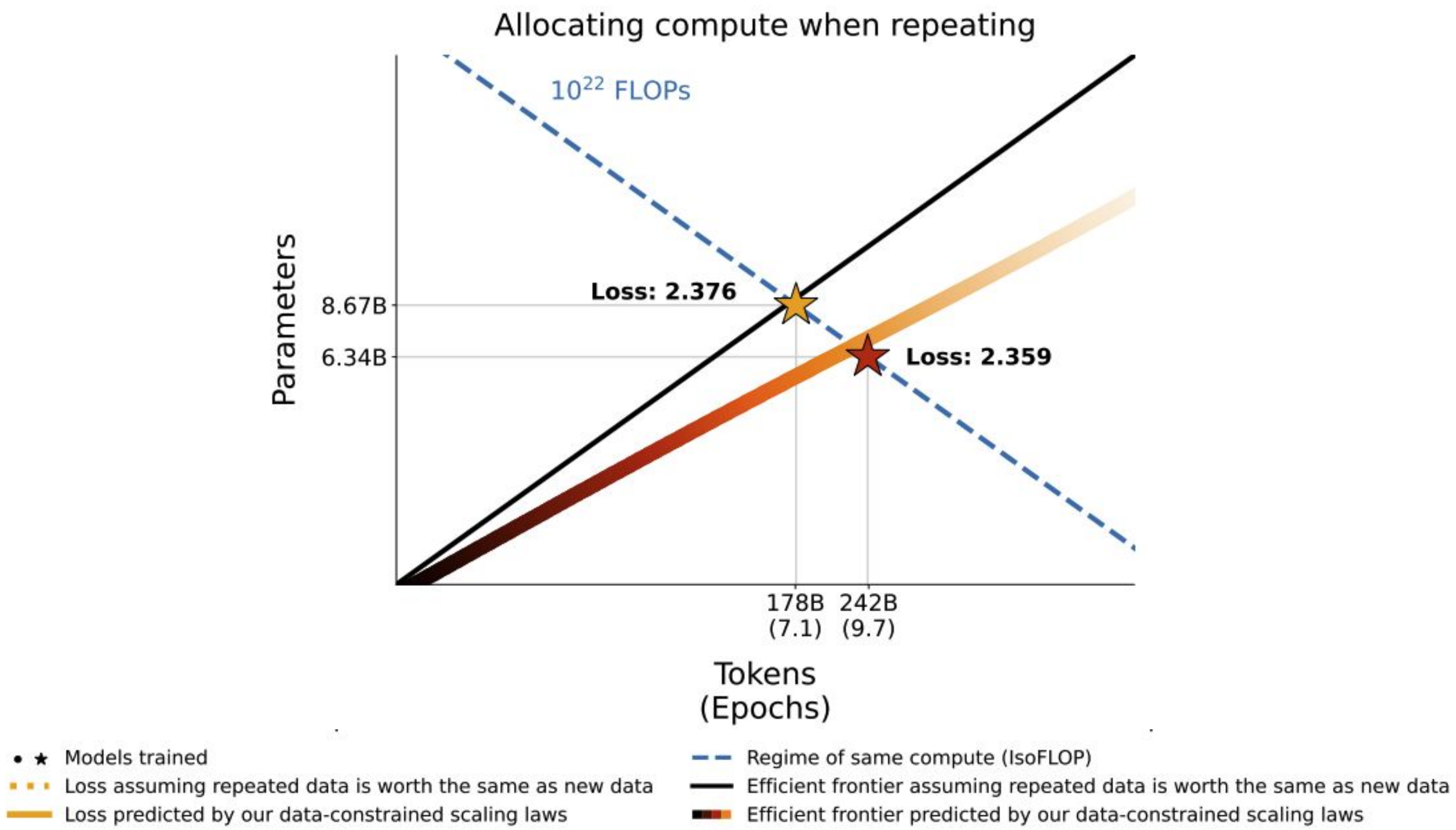
Fitting our model

$$R_D^* \sim= 15.4$$
$$R_N^* \sim= 5.3$$

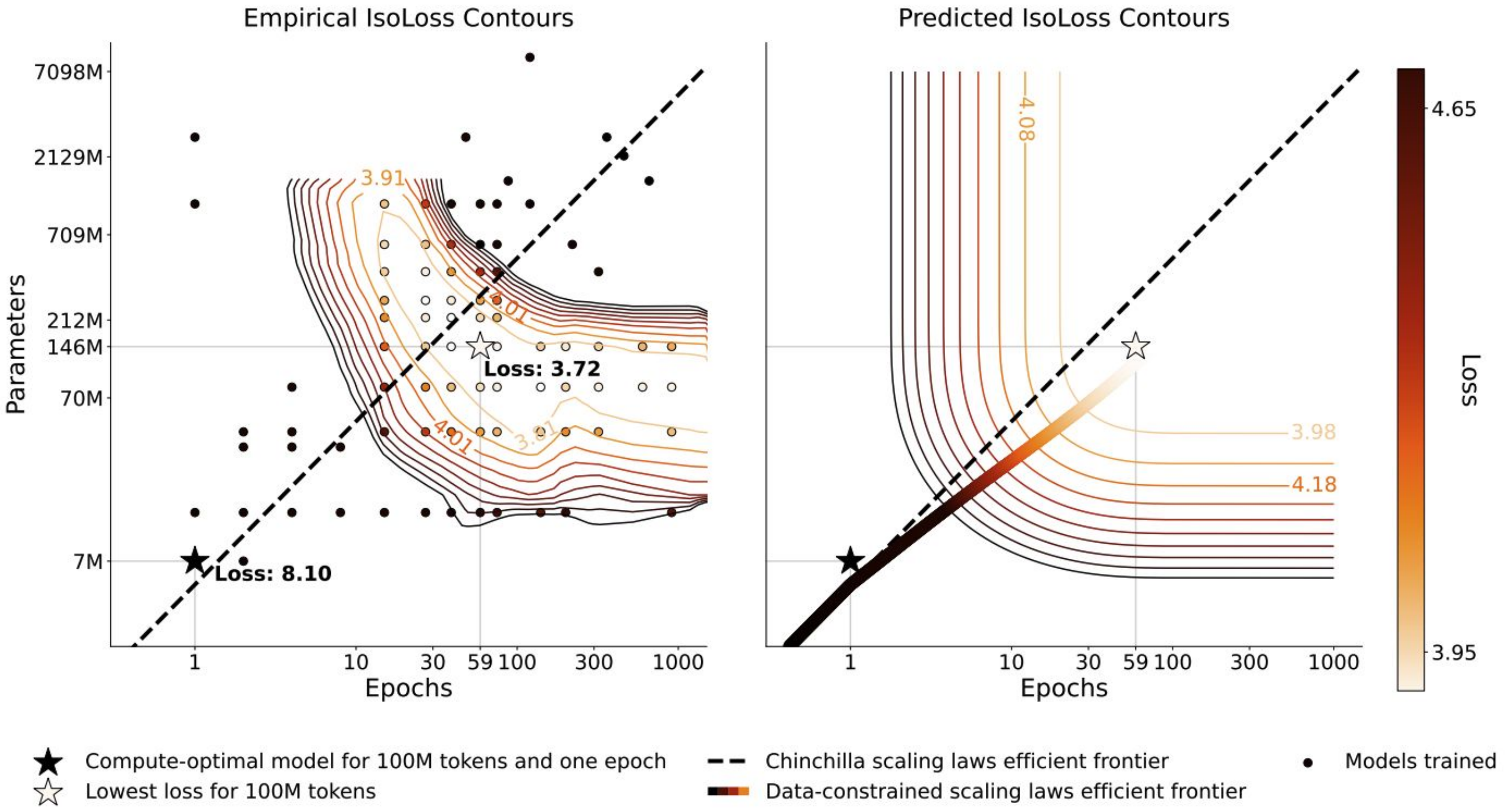
15 epochs before we see rapidly diminishing returns.
5x larger model before we see rapidly diminishing returns.

Suggests scaling epochs quicker than model size.

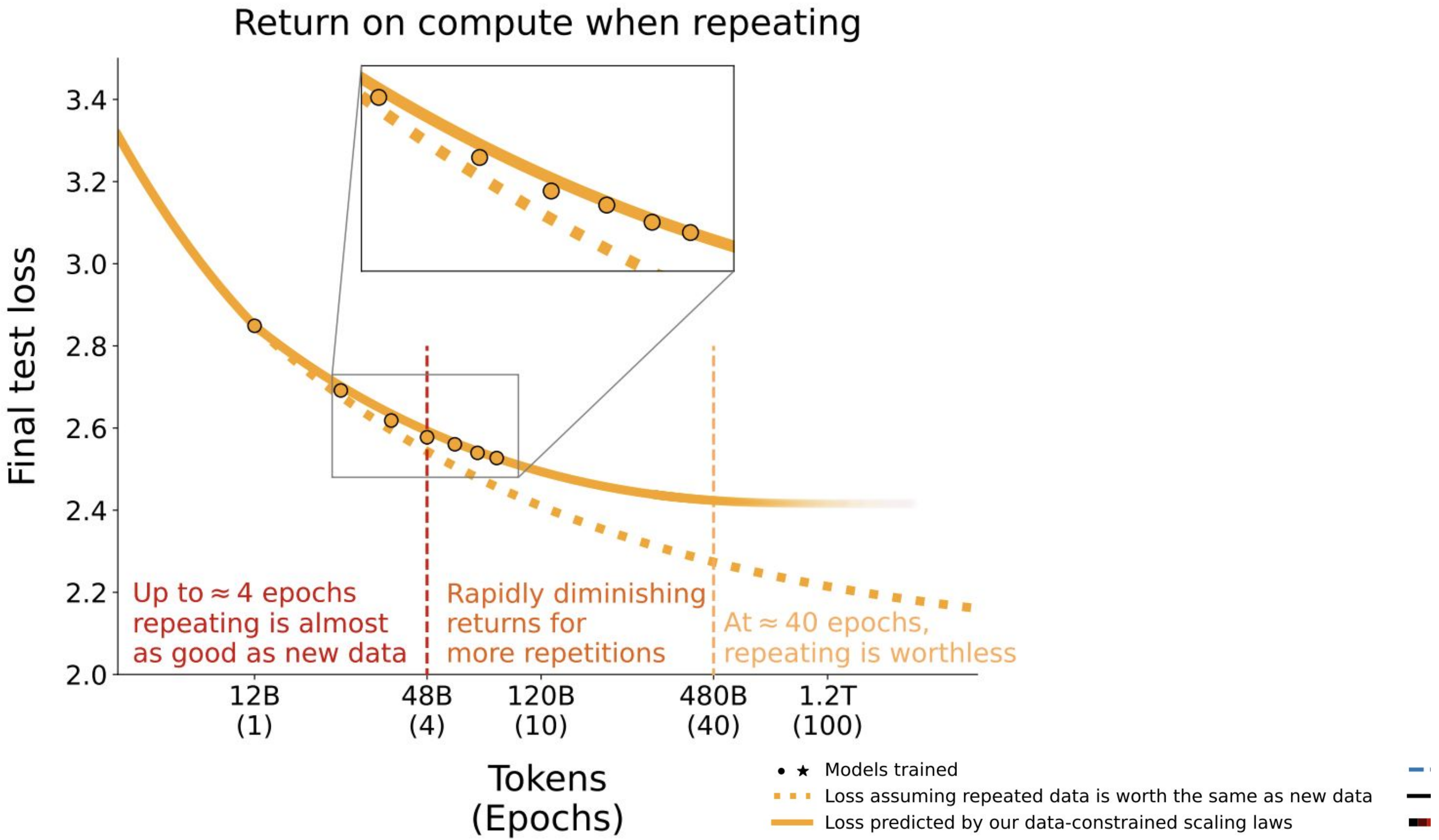
Experiments: Fixed Compute Budget



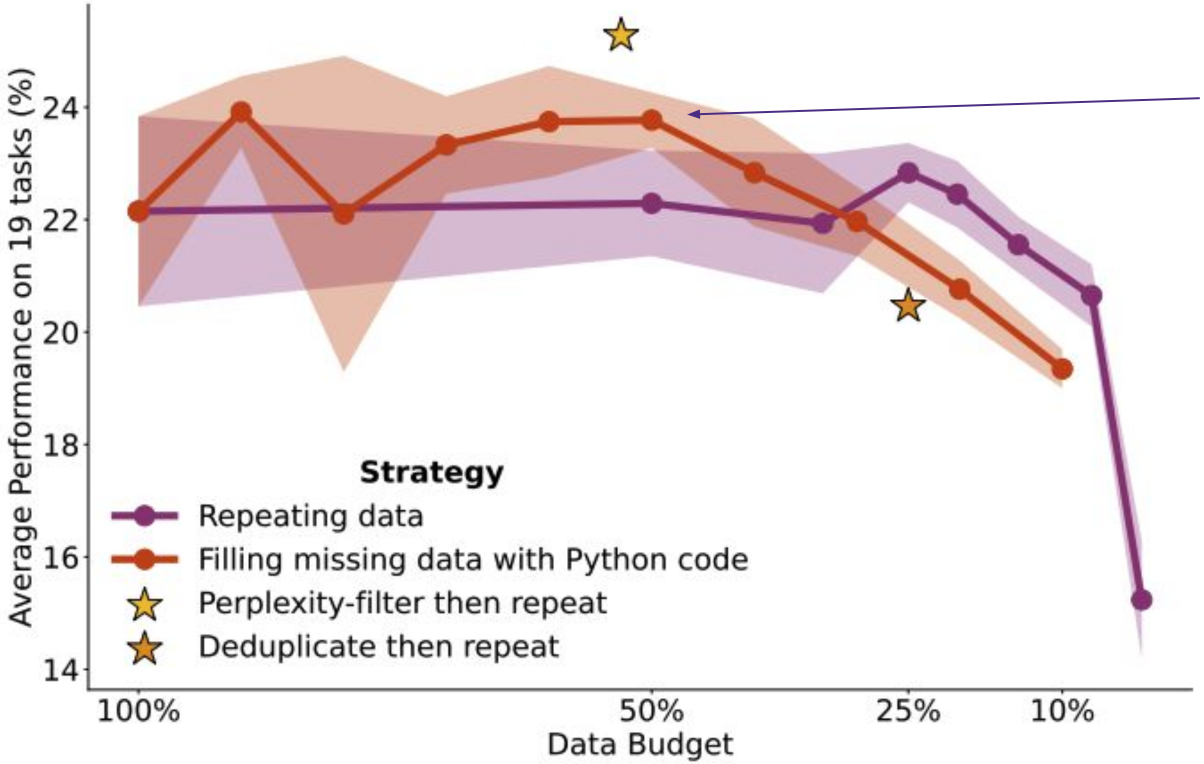
Experiments: Fixed Data Budget



Experiments: Return on scaling



Alternative strategy: Data augmentation



Having up to 50% of your data be code doesn't hurt performance!

Performance across tasks, not perplexity/loss

Impact

We have effectively 8x more data:

> Double dataset size by adding code
> Repeat for 4 epochs

…and more gains possible if you keep training.

> what about about memorization…?

Impact

FinGPT: Large Generative Models for a Small Language [14]

Risto Luukkonen †* Ville Komulainen † Jouni Luoma † Anni Eskelinen †
Jenna Kanerva † Hanna-Mari Kupari † Filip Ginter † Veronika Laippala †
Niklas Muennighoff ‡ Aleksandra Piktus ‡ Thomas Wang ‡ Nouamane Tazi ‡
Teven Le Scao ‡ Thomas Wolf ‡ Osma Suominen ◦ Samuli Sairanen ◦
Mikko Merioksa ◦ Jyrki Heinonen ◦ Aija Vahtola ◦ Samuel Antao °
Sampo Pyysalo †*
† TurkuNLP Group, University of Turku ‡ Hugging Face
◦ National Library of Finland ° AMD
*risto.m.luukkonen@utu.fi, sampo.pyysalo@utu.fi

OCTOPACK: INSTRUCTION TUNING CODE LARGE LANGUAGE MODELS

Niklas Muennighoff Qian Liu Armel Zebaze Qinkai Zheng Binyuan Hui
Terry Yue Zhuo Swayam Singh Xiangru Tang Leandro von Werra Shayne Longpre
n.muennighoff@gmail.com

[15]

SILO LANGUAGE MODELS: ISOLATING LEGAL RISK IN A NONPARAMETRIC DATASTORE

[16]

Sewon Min[*1] Suchin Gururangan[*1] Eric Wallace[2]
Hannaneh Hajishirzi[1,3] Noah A. Smith[1,3] Luke Zettlemoyer[1]
[1]University of Washington [2]UC Berkeley [3]Allen Institute for AI
{sewon,sg01,hannaneh,nasmith,lsz}@cs.washington.edu ericwallace@berkeley.edu

TinyLlama: An Open-Source Small Language Model

Peiyuan Zhang* Guangtao Zeng* Tianduo Wang Wei Lu
StatNLP Research Group
Singapore University of Technology and Design
{peiyuan_zhang, tianduo_wang, luwei}@sutd.edu.sg
guangtao_zeng@mymail.sutd.edu.sg

[17]

Perhaps a “default setting” in the future?

References

1. T. Brown, B. Mann, ..., and H. Lin, editors, Advances in Neural Information Processing Systems, volume 33, pages 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb49674 18bfb8ac142f64a-Paper.pdf.
2. O. Lieber, O. Sharir, B. Lenz, and Y. Shoham. Jurassic-1: Technical details and evaluation. White Paper. AI21 Labs, 2021.
3. J. Rae, S. Borgeaud, ..., and G. Irving. Scaling language models: Methods, analysis & insights from training Gopher. arXiv 2112.11446, 2021.
4. S. Smith, M. Patwary, ..., and B. Catanzaro. Using Deepspeed and Megatron to Train Megatron- turing NLG 530b, A Large-Scale Generative Language Model. arXiv preprint arXiv:2201.11990, 2022.
5. R. Thoppilan, D. D. Freitas, ..., and Q. Le. LaMDA: Language models for dialog applications, 2022.
6. E. Strubell, A. Ganesh, and A. McCallum. Energy and Policy Considerations for Deep Learning in NLP, 2019.
7. J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei. Scaling laws for neural language models, 2020.
8. J.Hoffmann, S. Borgeaud, A. Mensch, ..., and L. Sifre, Training Compute-Optimal Large Language Models, 2022.
9. R. Anil, A. Dai, ..., and Y. Wu, PaLM 2 Technical Report. arXiv preprint arXiv:2305.10403, 2023.
10. Nostalgebraist, LessWrong, chinchilla's wild implications, 2022. URL https://www.lesswrong.com/posts/6Fpvch8RR29qLEWNH/chinchilla-s-wild-implications.
11. P. Villalobos, J. Sevilla, L. Heim, T. Besiroglu, M. Hobbhahn, and A. Ho. Will we run out of data? An analysis of the limits of scaling datasets in Machine Learning. arXiv preprint arXiv:2211.04325, 2022.
12. Kudugunta S, Caswell I, Zhang B, Garcia X, Choquette-Choo CA, Lee K, Xin D, Kusupati A, Stella R, Bapna A, Firat O. Madlad-400: A multilingual and document-level large audited dataset. arXiv preprint arXiv:2309.04662. 2023.
13. Taylor R, Kardas M, Cucurull G, Scialom T, Hartshorn A, Saravia E, Poulton A, Kerkez V, Stojnic R. Galactica: A large language model for science. arXiv preprint arXiv:2211.09085. 2022.
14. Luukkonen R, Komulainen V, Luoma J, Eskelinen A, Kanerva J, Kupari HM, Ginter F, Laippala V, Muennighoff N, Piktus A, Wang T. FinGPT: Large Generative Models for a Small Language. In The 2023 Conference on Empirical Methods in Natural Language Processing 2023.
15. Muennighoff N, Liu Q, Zebaze A, Zheng Q, Hui B, Zhuo TY, Singh S, Tang X, Von Werra L, Longpre S. Octopack: Instruction tuning code large language models. arXiv preprint arXiv:2308.07124. 2023.
16. Min S, Gururangan S, Wallace E, Shi W, Hajishirzi H, Smith NA, Zettlemoyer L. SILO Language Models: Isolating Legal Risk In a Nonparametric Datastore. In NeurIPS 2023 Workshop on Distribution Shifts: New Frontiers with Foundation Models 2023.
17. Zhang, Peiyuan, Guangtao Zeng, Tianduo Wang and Wei Lu. TinyLlama: An Open-Source Small Language Model. arXiv preprint arXiv:2401.02385, 2024.
18. Hernandez D, Brown T, Conerly T, DasSarma N, Drain D, El-Showk S, Elhage N, Hatfield-Dodds Z, Henighan T, Hume T, Johnston S. Scaling laws and interpretability of learning from repeated data. arXiv preprint arXiv:2205.10487. 2022.
19. X. Bi, D. Chen, ..., and Y. Zhou. DeepSeek LLM: Scaling Open-Source Language Models with Longtermism, 2024.

Empirically, we have repeated some 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

Table 2.2 from [1], showing the training mix used for GPT-3

Empirically, little overfitting

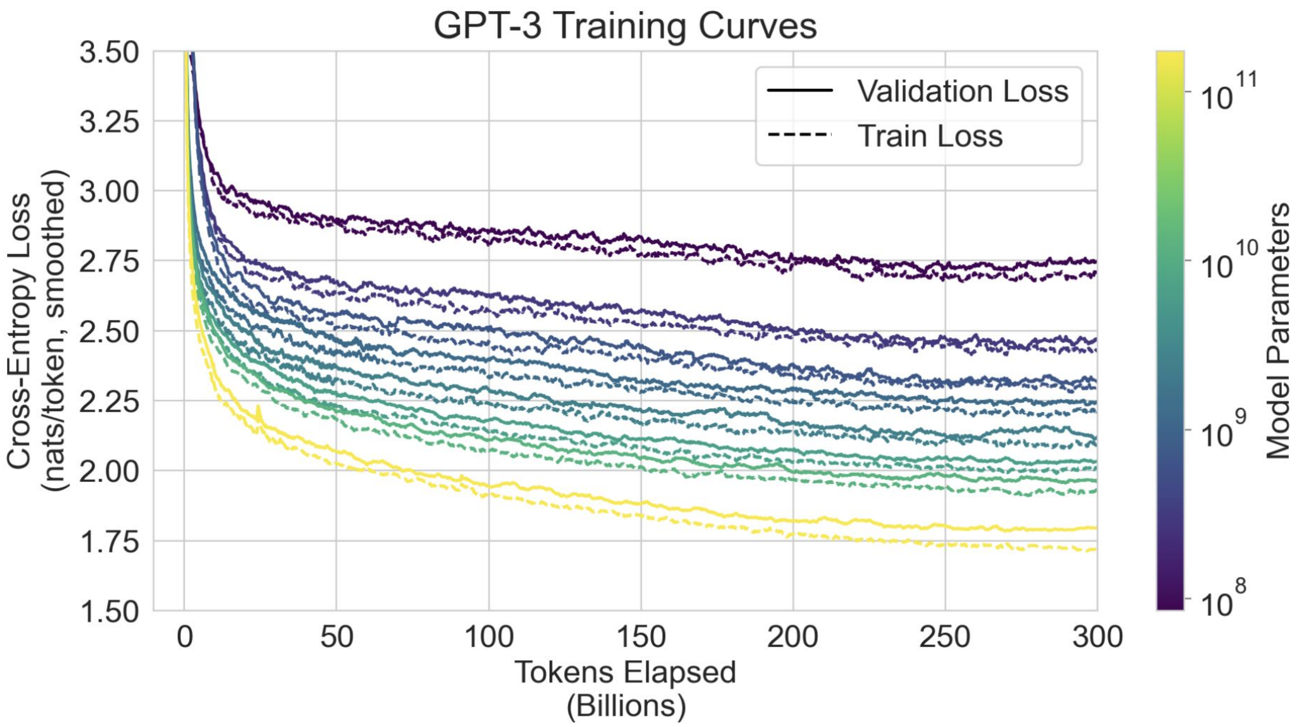


Figure 4.1 from [1]

Modelling repeated data

We define $$R_D^* = \frac{1-\delta}{\delta}$$

Why? So as R_D goes to infinity, D' goes to $U + R_D^* U$.

$$D' = U + (1-\delta)U\,\frac{(1-(1-\delta)^{R_D})}{\delta}$$

We assume δ is small, and get two approximations:

$$1/R_D^* = \frac{\delta}{1-\delta} \approx \delta \qquad e^{-\delta} \approx 1 - \delta$$

Modelling repeated data

We assume δ is small, and get two approximations:

$$1/R_D^* = \tfrac{\delta}{1-\delta} \approx \delta \qquad e^{-\delta} \approx 1 - \delta$$

Therefore:

$$(1 - \delta) \approx e^{-\delta} \approx e^{-1/R_D^*}$$

Recall:

$$R_D^* = \frac{1-\delta}{\delta}$$

And now we can directly modify our original equation:

$$D' = U + (1 - \delta)U\frac{(1-(1-\delta)^{R_D})}{\delta} = U + U \cdot R_D^* \cdot (1 - e^{-R_D/R_D^*})$$

What do we fit on?

Training compute (FLOPs)	Model size (# parameters)	Training data (# tokens)
9.3e20	2.8B	55B
2.1e21	4.2B	84B
9.3e21	8.7B	178B

For each setup, train 8 models with different amounts of unique training data that is repeated

\+ ~300 miscellaneous runs

All GPT-2-style decoder-only models with cosine LR decay. No early stopping. Using C4.
Figure from Sasha Rush's talk on the paper (https://www.youtube.com/watch?v=Kp5R6GZh8O0)

What do we fit on?

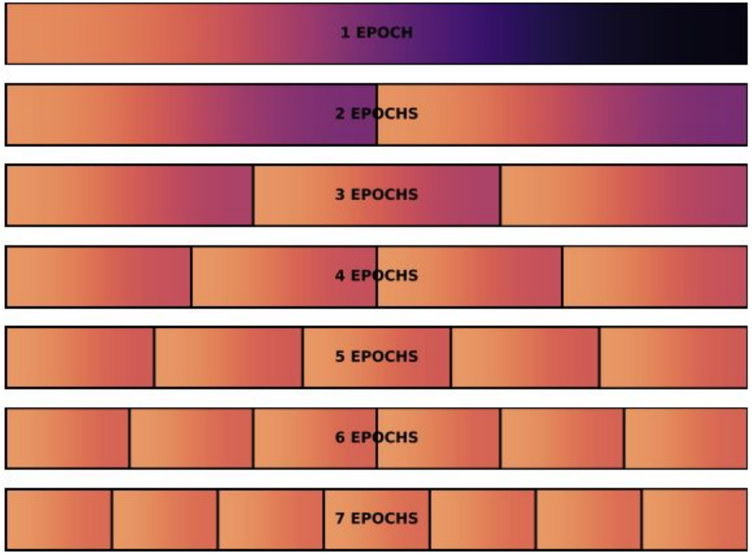
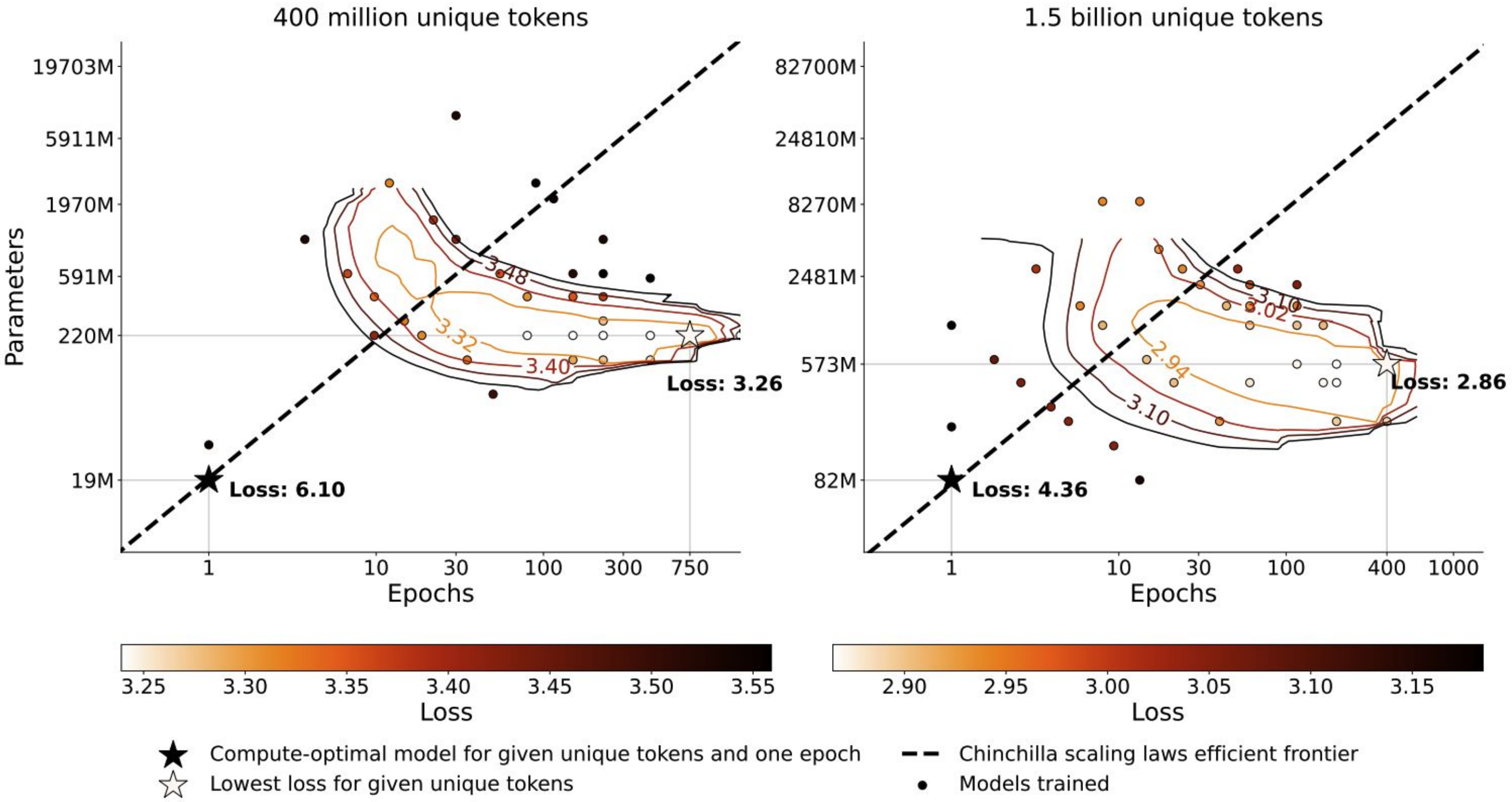


Figure 2: **Dataset setup.** We ensure that runs using less data (more epochs) always use a subset of the data used in runs with more data (fewer epochs).

Experiments: Fixed Data Budget



Fitting our model

We set U_N as the optimal number of parameters for U_D and our compute cost based on the chinchilla laws.

$$U_N = \min\{((U_D \cdot G)^{\beta/\alpha}) \cdot G, N\} \quad \textbf{where} \quad G = \left(\frac{\alpha A}{\beta B}\right)^{\frac{1}{\alpha+\beta}}$$

We then find *A*, *B*, *α*, *β* by fitting on the original chinchilla laws. This is done on experiments on C4 and gives:

$$L(N, D) = 1.87 + \frac{521}{N^{0.353}} + \frac{1488}{D^{0.353}}$$