[Paper Review]
Training Compute-Optimal Large Language Models

Paul Jason Mello
Department of Computer Science and Engineering
University of Nevada, Reno
pmello@unr.edu

Abstract

"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 $4\times$ more 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." [2]

1 Summary

This pivotal paper, released in 2022, redefined the training landscape of Large Language Models (LLMs). They pose the question "Given a fixed FLOPs budget, how should one trade-off model size and the number of training tokens?"[2] Through their experimentation, they demonstrate optimally trained models, like Chinchilla 70B will significantly outperform larger models like Gopher 280B with a fixed FLOPs budget by training on the optimal number of tokens. They go on to derive a training formula for the optimal-compute-data-parameter for any given training run.

2 Introduction

In 2022, frontier labs and companies had been in a race to train ever larger LLMs. This race was brought on by an important paper which proposed Neural Scaling Laws [3], which was itself built on prior works and observations detailing the same sparks. These papers essentially proposed ever larger models with ever larger datasets. Despite this, flaws and misunderstandings with the original studies produced inefficient models. With these growing models, single training runs have exploded in costs into the $10 - 100$s of millions for a single model. It is then crucial to ensure the model and training process is as efficient as possible. To determine this, the authors of this paper consider three main components which contribute to training efficiency, compute budget, data quantity, and parameter count.

3 Background and Motivation

Training ever larger models requires significant energy costs which increase with model size. Since training runs for LLMs can only be done once, it is important to ensure a near optimal training run. Prior works like Kaplan et. al [3] have shown a power law relationship between parameter count and performance where every $10\times$ increase in computational budget should be met with a $5.5\times$ increase in model size, and a training token increase of $1.8\times$. In this paper, the authors argue that modern LLMs have been severely starved of training data resulting in a hindrance of performance. Instead, increasing model size and training tokens in equal proportions leads to Chinchilla like models. Where Chinchilla, a 70B parameter model, significantly outperforms Gopher, a 280B parameter model, by a wide margin on a wide range of tasks including Measuring Massive Multitask Language Understanding (MMLU) [1].

3.1 Key Concepts

- **Concept 1:** "Given a fixed FLOPs budget, how should one trade-off model size and the number of training tokens?"[2].
- **Concept 2:** To model concept one, they define the final pre-training loss $L(N, D)$ as a function of model parameters N, number of training tokens D, and seek to find the optimal computational training budget C such that FLOPs$(N, D) = C$ where:

$$N_{\text{optimal}}(C), D_{\text{optimal}}(C) = \arg \min_{N,D \text{ s.t. FLOPs}(N,D)=C} L(N, D). \tag{1}$$

- **Concept 3:** They define parametric loss curves under the constraints defined in Kaplan et al.[3], FLOPs$(N, D) = 6ND$. This allows them to define the following power law form for optimality:

$$N_{\text{opt}}(C) = G\left(\frac{C}{6}\right)^{a}, \quad D_{\text{opt}}(C) = G^{-1}\left(\frac{C}{6}\right)^{b}, \text{where:} \tag{2}$$

$$G = \left(\frac{\alpha A}{\beta B}\right)^{\frac{1}{\alpha+\beta}}, \quad a = \frac{\beta}{\alpha+\beta}, \quad \text{and} \quad b = \frac{\alpha}{\alpha+\beta}. \tag{3}$$

  Here, a and b define a roughly $50\% - 50\%$ split on N and D as described in the following section.
- **Concept 4:** Utilizing concept two, they illustrate that models can be trained optimally to reduce loss under the constraints like compute budget, data quantity, or parameters resulting in better model capabilities. In other words, training optimally dense models on optimal tokens leads to optimal models.

4 Methodology

To test their hypothesis, they devise an empirical analysis of 400 language models consisting of 70M to 16B parameters and train them on 5B - 500B tokens. They utilize their pre-training loss hypothesis to predict optimal bounds and calculate the necessary number of tokens to optimally train a model.

4.1 Overview of the Proposed Approach

- **Issue 1:** Given the exceptionally large and long training runs of larger models, they test their hypothesis on various models at a significantly smaller scale and extrapolate their findings to larger models.
- **Issue 2:** Prior work from Kaplan et al.[3], described scaling laws for the optimal scaling of N and D to be a roughly $75\% - 25\%$ split respectively. However, in this paper, the authors derive a mathematically optimal scaling law for N and D to be a roughly $50\% - 50\%$ split.
- **Issue 3:** Two key assumptions are made and one important assumption ignored. The first is that their test models, despite significantly lower parameter counts, follow the same scaling curve. The second assumption assumes that this methodology can be inherently comparative, as it measures its performance alongside other models. The final and unmentioned

assumption is that all data is inherently equal in quality, and that is fundamentally not the case. As a result, token counts will likely vary depending on quality.

5 Experiments and Results

They conduct their experiments across a wide range of models with varying parameter counts and training tokens. They provide what is essentially a look up table / equation defining the estimated optimal training parameters to produce the most performant models.

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

Figure 1: Estimated optimal training parameters for a given compute budget.

Where as LLMs had previously been trained with the following:

Model	Size (# Parameters)	Training Tokens
LaMDA (Thoppilan et al., 2022)	137 Billion	168 Billion
GPT-3 (Brown et al., 2020)	175 Billion	300 Billion
Jurassic (Lieber et al., 2021)	178 Billion	300 Billion
Gopher (Rae et al., 2021)	280 Billion	300 Billion
MT-NLG 530B (Smith et al., 2022)	530 Billion	270 Billion
Chinchilla	70 Billion	1.4 Trillion

Figure 2: 5 of the largest dense transformer language models with their parameter counts and training tokens. Chinchilla illustrates a common misconception in the necessary quantity of training data that has been pervasive in prior models.

5.1 Evaluation Metrics

- **Evaluation Task 1:** They evaluate Chinchilla across a wide range of tasks comprised of Language Modelling, MMLU, BIG-bench, Common Sense, Question Answering, and Reading Comprehension.
- **Metric 1:** They compare Chinchilla against a range of LLMs and show Chinchilla directly outperforms Gopher metrics like perplexity, bits-per-byte, and inference speed/cost.

5.2 Key Results

- **Result 1:** Prior training models were starved of necessary tokens to generate highly performant models. Larger models consisting of 175B parameters were trained on 300B tokens when they should have been trained on 3.7 trillion with a compute budget of roughly $1e^{24}$ FLOPs
- **Result 2:** Chinchilla, while $4\times$ smaller than Gopher, and other large models became SOTA with an MMLU benchmark accuracy of 67.5% and outperformed on Gopher by 7%. A focus is made on MMLU because training data leakage is an important component.

6 Discussion and Critique

Overall, this paper defines a pivotal moment of training LLMs. It illustrates common misconceptions that frontier labs and companies failed to identify the starvation of modern models with insufficient token counts. They propose a novel algorithm to identify the necessary tokens and training parameters given a fixed compute budget and explore their algorithm by aggregating and training smaller models. They demonstrate that optimal models, such as Chinchilla, which was trained according to their algorithm, have significant performance gains over larger models like MT-NLG 530B and GPT-3 on common evaluation tasks like MMLU.

Despite this paper, or better yet because of this paper, models that are released today are significantly decreasing across N, D, and C. Despite this, these models are better performing and significantly more cost efficient which will become ever more important as they are integrated into a range of mobile or resource limited devices beyond the extensive GPU warehouses they exist in now. As of today, only a few labs and companies, such as Anthropic, OpenAI, Google, Meta, and maybe Mixtral have the funding, compute, data, and know-how to train foundation models exceeding 50B parameters.

6.1 Strengths

- **Strength 1:** Improvements over prior works like Kaplan et al.[3]. Illustrating research gaps understanding the proper training paradigms for LLMs exceeding 1M parameters.
- **Strength 2:** The applicability of this approach directly leads to better scalability, better generalization, better models, and ultimately more efficient training/inference/and energy costs.
- **Strength 3:** This paper introduced a significant shift in conceptions regarding the training of ever larger LLMs and did so using smaller models with theoretical justifications for larger scales. Efficient in both compute and theory.

6.2 Weaknesses

Notably, this paper was produced by Google's DeepMind. There are no important weaknesses as they address the scope of their work from the beginning and propose novel improvements to the work proposed in Kaplan et al.[3]. However, as described earlier not all tokens are created equal. This is a systemic issue with all datasets, however a cleaner dataset would likely provide marginal improvements to the performance gained or required tokens during training. However, since most models are trained on roughly the same data, token quality should not significantly effect the results of their work.

7 Future Directions

- One direction would be to significantly clean the datasets and tokens used during training to provide a more thorough understand of the role of N, D, and C, for the efficient computation LLMs.
- Although significantly inefficient, double checking the results of this work on a wide range of language models beyond 70B parameters would be important to ensure that the optimal N, D, and C holds for models of larger sizes rather than relying on the proposed power scaling laws. This would be a grossly inefficient evaluation across a range of dimensions like memory, compute usage, carbon footprint, and more, but one can never be certain.

8 Conclusion

The authors introduces a compute-optimal training approach for LLMs, demonstrating that scaling both model size and training tokens in equal proportions leads to better performance. They train Chinchilla, a 70B parameter model, using $4\times$ more data than Gopher, a 280B parameter model, resulting in superior performance on key benchmarks despite being $4\times$ smaller. This work highlights the efficiency gains of optimal model and data sizes, effectively reducing the training and downstream inference compute. As a result, they have promoted more sustainable and compute-efficient

model development. Today this has become a landmark paper for the training of extremely large LLMs and has seen usage to optimally train models of significantly smaller sizes, in the millions of parameters.

References

[1] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding, 2021.

[2] 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. Training compute-optimal large language models, 2022.

[3] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020.