CS336 Assignment 3

Radostin Cholakov

May 2025

Problem `chinchilla_isoflops`.

Using the provided data, we can find the minimum loss achieved at each compute budget and extrapolate to the desired scale. We can predict:

- For compute budget `1e23`, predicted parameters around 50B (`5.00222576e+10`).
- For compute budget `1e24`, predicted parameters around 126B (`1.26757796e+11`).

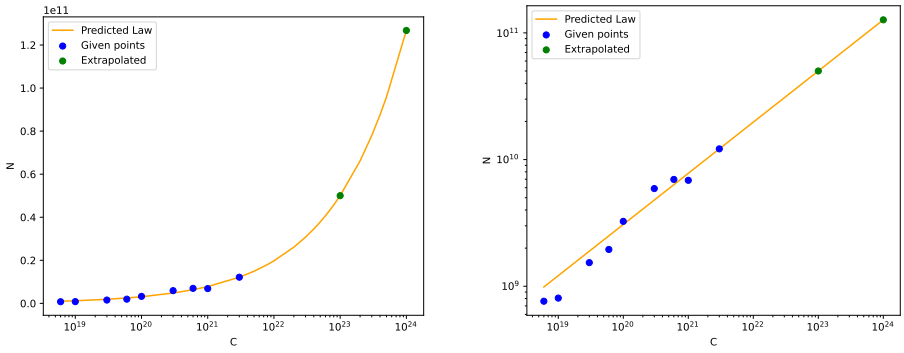

Figure 1: Optimal parameters. The log-log plot for our extrapolation is shown on the right. On the left, the power law is plotted with linear y-axis to better demonstrate the impact of higher compute budgets.

Using our fitted model, we can derive the optimal data amount. Data-parameter relationship: $C = 6ND$, thus $D = C/6N$. We can predict:

- At budget `1e23`, we need around 333B tokens (`3.33185016e+11`).
- At budget `1e24`, we need around 1.3T tokens (`1.31484352e+12`).

Additionally, we can fit quadratic curves similar to IsoFLOPs [1] and use the smallest points of the quadratics.

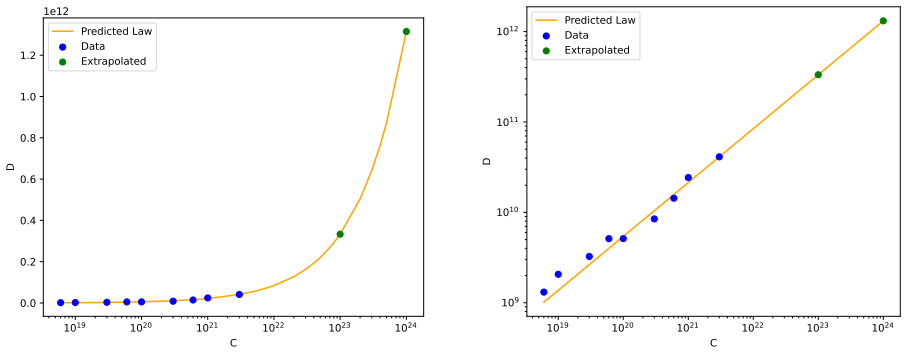

Figure 2: Optimal tokens. The log-log plot for our extrapolation is shown on the right. On the left, the power law is plotted with linear y-axis to better demonstrate the impact of higher compute budgets.

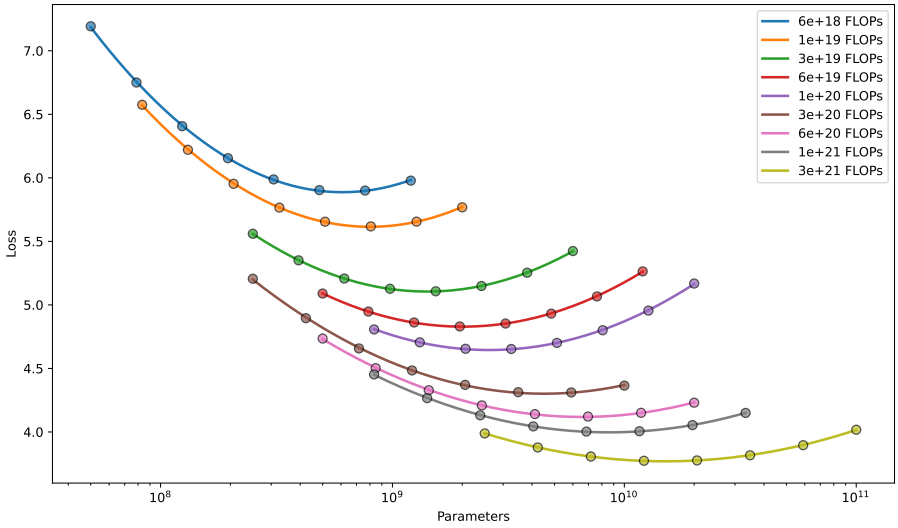

Figure 3: IsoFLOPs quadratic curves for the provided synthetic data.

Problem `scaling_laws`.
Following works in the field of scaling LLM training such as [1] and [2], the initial strategy to pursue is as follows:

- Certain aspect ratios of $\frac{d_{model}}{n_{head}}$ and $\frac{d_{model}}{n_{layer}}$ are quite robust to changes of the specific values that produce the ratio. This allows us to fix these ratios and only change one of the parameters to have lower degrees of freedom when performing hyperparameter search. Then, we can use the relationship $N = 12 \cdot n_{layer} \cdot d_{model}^2$ to derive the total number of params

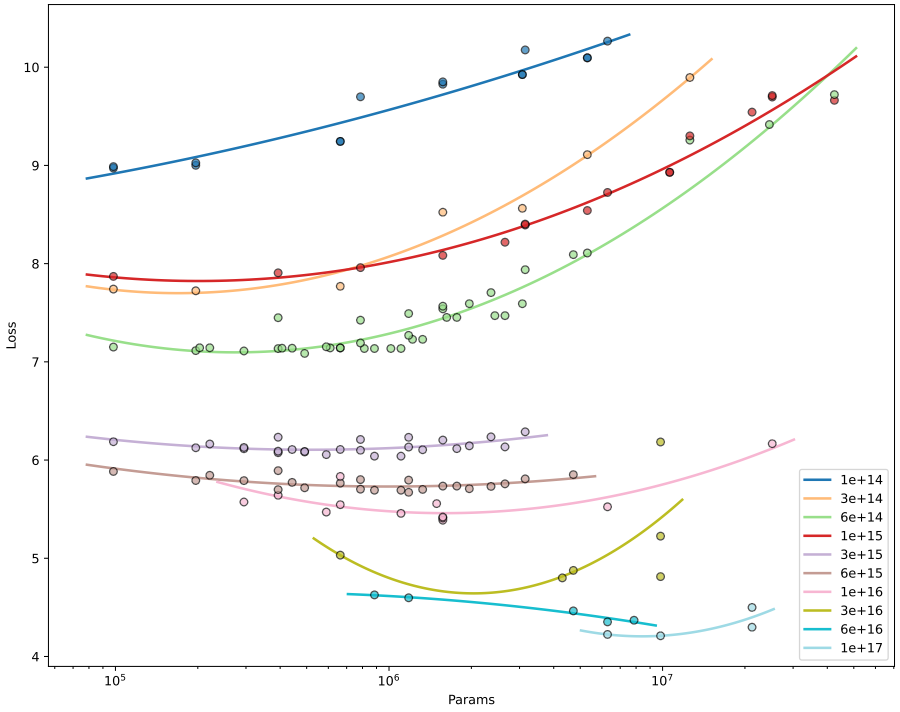

Figure 4: Aggregated isoFLOPs curves with my experiment logs: grid search + varying dmodel for given ratios.

used for measurements.

- From the used literature initial guesses are $\frac{d_{model}}{n_{head}} = 64$ and $\frac{d_{model}}{n_{layer}} = 64$, however, I also find that for smaller model sizes better ratios are 16 and 32.
- Smaller batch sizes are usually better because they allow us to perform more gradient steps. The API allows batch sizes 128 and 256, so we naturally go with 128. A few experiments were still performed with all parameters set equal except for batch size and 128 performed best.
- Since a batch size of 128 is stil high enough to perform stable training, I pick learning rate `1e-3` as my initial choice. Also, I follow the logic that a good learning rate might be close to divergence and I observe that for all my experiments divergence is probably with learning rates above `1e-3`.

I decided to run a big sample of experiments for smaller flop budgets and make educated guesses about how to update the layer and head aspect ratios as I increase the flop budgets. The initial observation at sizes `1e13` to around `3e14` is

that I get quite noisy losses which do not change predictably with increasing or decreasing the model size. Also, perhaps at this scale, small enough model sizes cannot be reached because the smallest combination of parameters is 2 layers and `dmodel` 64, so the smallest model is 98k and we cannot observe meaningful isoFLOPs-like results.

The next experiments will be performed on model sizes from `1e14` to `1e17` with sweeps of hyperparameters with aspect ratios for layers and heads of 64, 32, and 16. Additionally, I decided to run a grid search on the hyperparameters for cheaper flop budgets: up to `3e15`. The values for these sweeps are as follows: `dmodel=64, 96, 128`, `num_layers=2, 4, 6, 8, 10, 12, 16, 20, 24`.

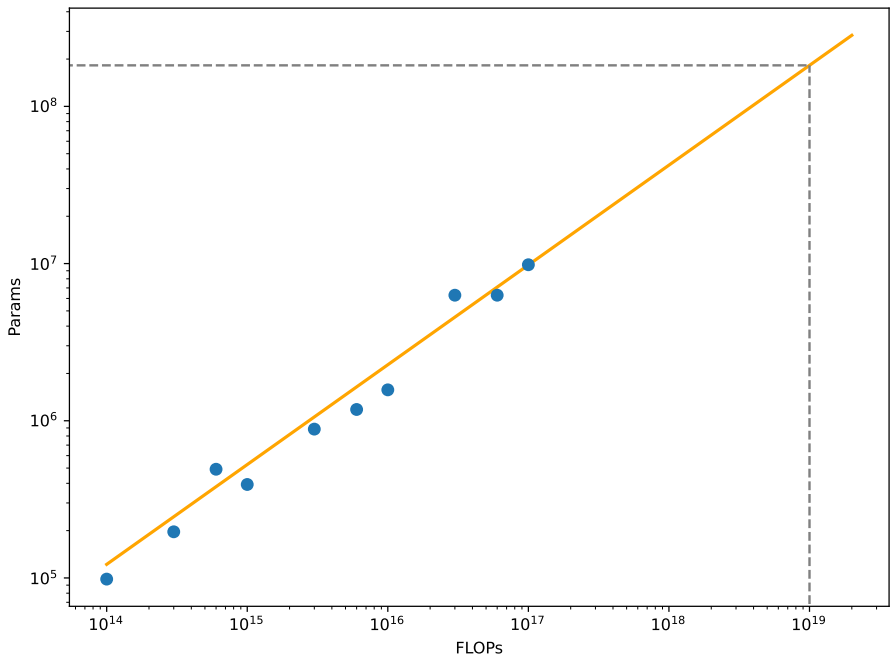

Figure 5: Parameter count projection. Fitted scaling law with best performing models at each flops budget.

Using all the observations with fixed aspect ratios as well as grid search for lower flop budgets, I aggregated the data and for my final projections used the best achieved loss for each budget to fit a power law. Initially, when I did not have enough data points, my projections were quite unstable and each new data point changed the projected parameter count in the range from tens of millions to tens of billions. Later, with more data points collected, my parameter projections were more stable – around 100 to 200 million parameters. I also used

the LLaMA 3 [3] paper as a reference, since it performed IsoFLOPs analysis at various data/parameter sizes. I believe that my final projection of around 180 million non-embedding parameters agrees with the order of magnitude discussed in the paper as well as other publications in recent days.

PREDICTED PARAMETERS:
The projected parameter count I can calculate from my scaling law is 182,413,449. I observed that layer aspect ratios of 16 worked best for smaller models and 32 for bigger models. Aspect ratios of 64 and above gave poor results in my experiments. Solving for `d_model` with chosen ratios 32 and 32 for layers and heads, I get $2\sqrt[3]{60804483}$ which I will approximate to 768 for divisibility purposes.

`batch_size`: 128
`d_model`: 768
`num_layers`: `d_model` / 32 = 24
`num_head`: `d_model` / 32 = 24
`lr`: `1e-3`

PREDICTED LOSS: Fitting the same power law but for losses instead of parameter counts yields a predicted loss of `2.494`.

I provide code for fitting scaling laws (prepared for Q1 but also used for Q2).

References

[1] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

[2] 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. *arXiv preprint arXiv:2001.08361*, 2020.

[3] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

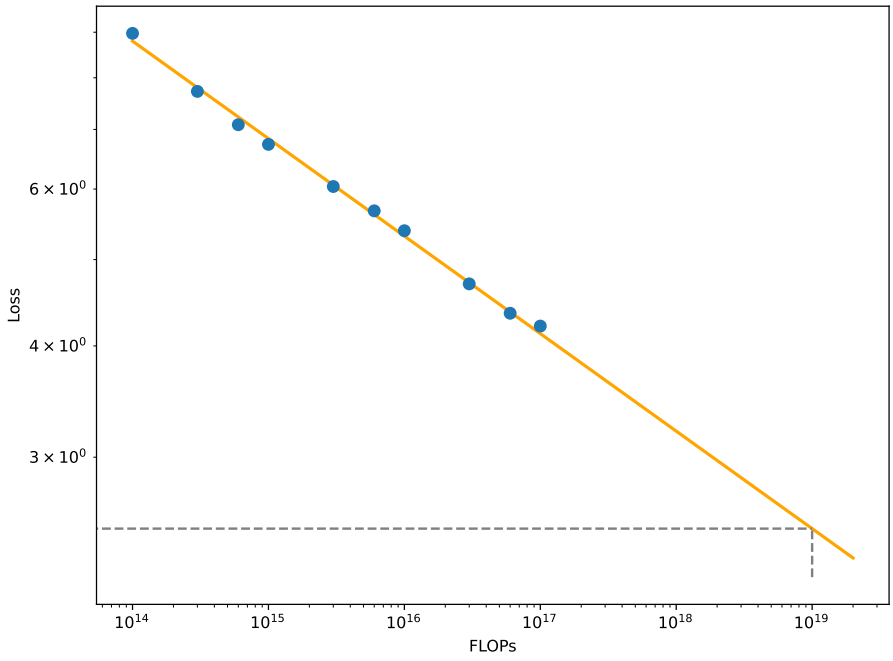

Figure 6: Loss projection. Uses fitted scaling law with best performing models at each flops budget. The fit looks very good but I believe we will see some deviation to the right with higher compute budgets.