CS336 Assignment #3 (scaling): Scaling Laws

Yin Xiaogang

Spring 2025

If you are using LaTeX, you can use `\ifans{}` to type your solutions.
Please tag the questions correctly on Gradescope, otherwise the TAs will take points off if you don't tag questions.

1. Scaling Laws Review (5 points)

(a) (5 points) Problem (chinchilla_isoflops)
Write a script to reproduce the IsoFLOPs method describe above for fitting scaling laws using the final training loss from a set of training runs. For this problem, use the (synthetic) data from training runs given in the file data/isoflops_curves.json. This file contains a JSON array, where each element is an object describing a training run. Here are the first two runs for illustrating the format:

```
[
    {
    "parameters": 49999999,
    "compute_budget": 6e+18,
    "final_loss": 7.192784500319437
    },
    {
    "parameters": 78730505,
    "compute_budget": 6e+18,
    "final_loss": 6.750171320661809
    },
    ...
]
```

For fitting the scaling laws, the scipy package (and scipy.optimize.curve_fit in particular) might be useful, but youre welcome to use any curve fitting method youd like. While Hoffmann et al. [2022] fits a quadratic function to each IsoFLOP profile to find its minimum, we instead recommend you simply take the run with the lowest training loss for each compute budget as the minimum.

i. Show your extrapolated compute-optimal model size, together with the $\langle C_i, N_{opt}(C_i)\rangle$ points you obtained. What is your predicted optimal model size for a budget of 10^{23} FLOPs? What about for 10^{24} FLOPs?
Deliverable: A plot showing your scaling law for model size by compute budget, showing the data points used to fit the scaling law and extrapolating up to at least 10^{24} FLOPs. Then, a one-sentence response with your predicted optimal model size.
Solution: My predicted optimal model size for a budget of 10^{23} FLOPs is 39,792,556,129, and 85,491,178,611 for 10^{24} FLOPs.
I have obtained the following formula using scipy.optimize.curve_fit.

$$N = a \cdot C^b + c \qquad \text{a=1.350e+03, b=3.253e-01, c=-1.192e+09}$$
$$= 1350 \cdot C^{0.3253} - 1.192 \cdot 10^9$$

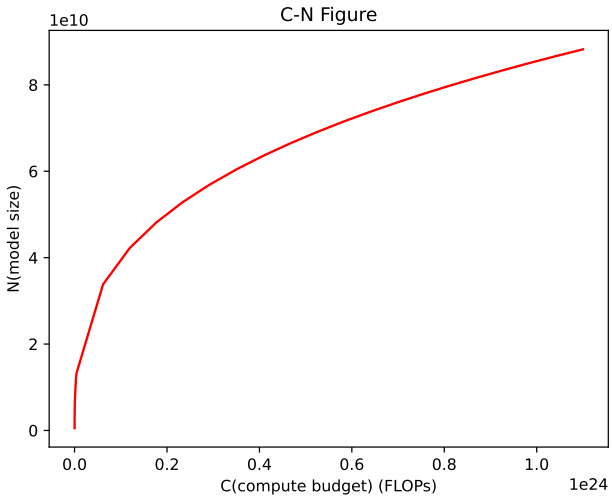

Figure 1: C-N figure

ii. Show your extrapolated compute-optimal dataset size, together with the $\langle C_i, D_{opt}(C_i)\rangle$ data points from the training runs. What is your predicted optimal dataset size for budgets of 10^{23} and 10^{24} FLOPs? **Deliverable**: A plot showing your scaling law for dataset size by compute budget, showing the data points used to fit the scaling law and extrapolating up to at least 10^{24} FLOPs. Then, a one-sentence response with your predicted optimal dataset size.

Solution: My predicted optimal model size for a budget of 10^{23} FLOPs is 364,796,370,806, and 1,525,768,382,027 for 10^{24} FLOPs.

I have obtained the following formula using scipy.optimize.curve_fit.

$$D = a \cdot C^b + c \qquad \text{a=1.802e-03, b=6.220e-01, c=5.948e+08}$$
$$= 0.001802 \cdot C^{0.6220} + 5.948 \cdot 10^8$$

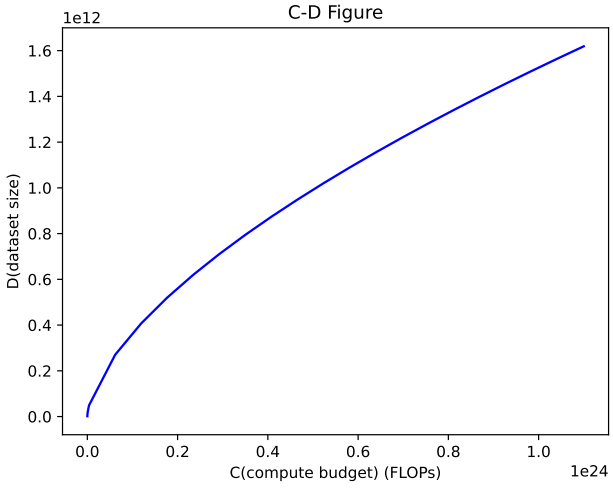

Figure 2: C-D figure

2. Constructing Scaling Laws (50 points)

(a) (50 points) Problem (scaling_laws)

Construct a scaling law to accurately predict the optimal model size, its hyperparameters, and the associated training loss for a FLOPs budget of 1e19. To construct your scaling laws, you will use our training API to query the final training loss for various experimental configurations (§3.1); you may not query more than 2e18 FLOPs worth of experiments for fitting your scaling law. This is hard cap that will be enforced by the API.

Deliverable: A typeset write-up that contains a complete description of your approach and methodology for fitting a scaling law. In addition, it should describe how you use the scaling law to predict the optimal model size for the given FLOPs budget, and your predicted values. The write-up should include commentary about why you made particular design decisions, and the description should be detailed enough to reproduce your approach and results.

Note on batch size: We place essentially no constraints on the hyperparameters you may report under the FLOPs budget of 1e19, other than the following requirement: **your batch size must be either 128 or 256.** This is done to ensure that runs have reasonably high model FLOPs utilization. If we have issues with out-of-memory errors when running your reported hyperparameter configuration, we will either use gradient accumulation or scale the number of data parallel GPUs to maintain your desired batch size.

To help you get started, we recommend thinking about at least the following questions. Your writeup should contain additional commentary about how decisions were made for each factor below:

- Given your fixed scaling laws budget of 2e18, how did you decide which runs to query?
- How did you fit your scaling law? Describe the concrete method or methods you used. In particular, it will likely to be useful to familiarize yourself with the approaches used in Kaplan et al.[1] and Hoffmann et al.[2]
- How well does your scaling law fit the experimental data?
- For our given FLOPs budget of 1e19, what optimal model size does your scaling law predict? What is the predicted loss?
- If you were to train a model with your predicted optimal number of parameters, what hyperparameters would you use? To estimate the number of non-embedding parameters for a given model hyperparameter configuration, use $12 n_{layer} d_{model}^2$.

In addition to the report, submit your (1) predicted optimal model size, (2) the training hyperparameters to use including either batch size 128 or 256, and (3) the models training loss to this Google form: `https://forms.gle/sAUSLwCUETew2hYN6`. Part of your grade on the assignment will be determined by the performance of your predicted optimal model.

Solution: I can not access the training api. To be done

References

[1] 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. arXiv:2001.08361.

[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. arXiv:2203.15556.

Submission Instructions

You shall submit this assignment on GradeScope as two submissions – one for "Assignment 2 [coding]" and another for 'Assignment 2 [written]":

1. Run the `collect_submission.sh` script to produce your `assignment2.zip` file.
2. Upload your `assignment2.zip` file to GradeScope to "Assignment 2 [coding]".
3. Upload your written solutions to GradeScope to "Assignment 2 [written]".