LLM Seminar 3

Scaling Laws and GPT-3

Yue Lin 林越 @ CUHKSZ - 2024.6.20

About Me

Yue LIN 林越

- About to be Prof. Baoxiang Wang's PhD student this fall.
- Personal Website & Blog: https://yuelin301.github.io/
- Research Interests:
    - Sequential Social Dilemma
        - Multi-Agent Reinforcement Learning
        - Algorithmic Game Theory (Especially in Information Design)
    - Large Language Models (Beginner Level)

GPT-1 (OpenAI. 2018. “Improving language understanding by generative pre-training.”)

1. GPT-1 = Decoder (in Transformer, with learnable positional encoding) + Pre-Training + Fine-Tuning.
2. Pre-Training: Unsupervised learning. The model is trained using unlabeled data to predict the next word. It is used to make the model to be familiar with human common knowledge.
3. Fine-Tuning: Supervised learning. The model is trained using labeled data for specific downstream NLP tasks.

GPT-2 (OpenAI. 2019. “Language models are unsupervised multitask learners.”)

1. GPT-2 = Decoder (in Transformer) + Pre-Training + Turning Fine-tuning to Pre-Training + More Parameters.
2. Enhanced pre-training. Eliminated fine-tuning.
    1. The unsupervised objective of the earlier pre-training is demonstrated to be the same as the supervised objective of the later fine-tuning.
    2. The downstream tasks can be reconstructed to be descripted in the form used in pre-training.
    3. A competent generalist is not an agregation of narrow experts.
3. The scaling law is initially emerging: The more parameters, the better the performance, and the improvement is very stable.
4. The Number of Parameters: 1.5B

Scaling Laws for Neural Language Models

OpenAI. 2020.1.

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

The Model

- “We primarily train decoder-only Transformer models, though we also train LSTM models and Universal Transformers for comparison.”
- GPT-1 is cited. Not GPT-2.

Conclusion 1

Scale vs. Model Shape

- Model performance depends most **strongly** on **scale**
    - the number of model parameters N
    - the size of the dataset D
    - the amount of compute C
- Within reasonable limits, performance depends very **weakly** on other **architectural hyperparameters**
    - Depth vs. width.
    - Multi-heads

Conclusion 2

Power-Law

- Performance has a **power-law** relationship with each of the three scale factors N, D, C when not bottlenecked by the other two.

In the given sentence, "power-law scalings" refers to a mathematical relationship where the relationship between certain variables can be described by a power-law function. Specifically, if two variables x and y satisfy the following relationship:

$$y \propto x^{\alpha}$$

where α is a constant, then we say that y and x follow a power-law relationship.

The Amount of Compute C

PF-Day

- PF = Petaflop = Peta Float Operation
  - 1 Peta = 10e15 = 一千万亿
- PF-Day is a unit to describe the amount of compute.
  - “1 PF-day” equals 10e15 floating-point operations per second, continuously for one day (24 hours).

- $C \approx 6NBS$ – an estimate of the total non-embedding training compute, where B is the batch size, and S is the number of training steps (ie parameter updates). We quote numerical values in PF-days, where one PF-day $= 10^{15} \times 24 \times 3600 = 8.64 \times 10^{19}$ floating point operations.

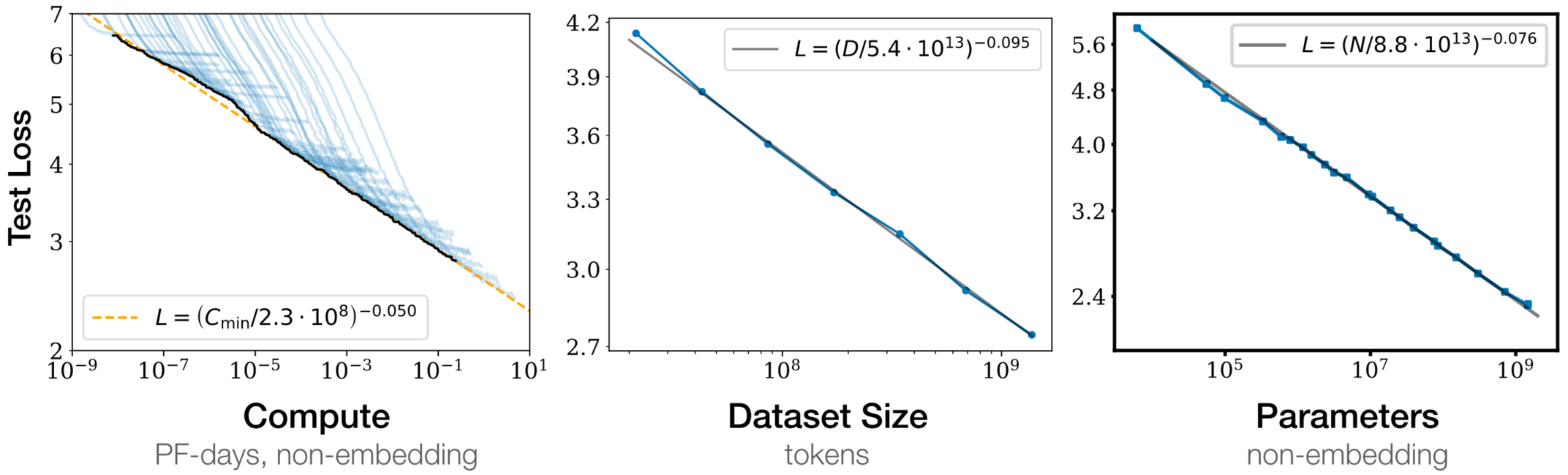


Figure 1 Language modeling performance improves smoothly as we increase the model size, datasetset size, and amount of compute[2] used for training. For optimal performance all three factors must be scaled up in tandem. Empirical performance has a power-law relationship with each individual factor when not bottlenecked by the other two.

- "We observe no signs of deviation from these trends on the upper end, though performance must flatten out eventually before reaching zero loss."

Parameters	Data	Compute	Batch Size	Equation
N	∞	∞	Fixed	$L(N) = (N_c/N)^{\alpha_N}$
∞	D	Early Stop	Fixed	$L(D) = (D_c/D)^{\alpha_D}$
Optimal	∞	C	Fixed	$L(C) = (C_c/C)^{\alpha_C}$ (naive)

- Appendix A. Table 4.
- $N_c, D_c, C_C, \alpha_N, \alpha_D, \alpha_C$ are constant.

Power Law	Scale (tokenization-dependent)
$\alpha_N = 0.076$	$N_c = 8.8 \times 10^{13}$ params (non-embed)
$\alpha_D = 0.095$	$D_c = 5.4 \times 10^{13}$ tokens
$\alpha_C = 0.057$	$C_c = 1.6 \times 10^{7}$ PF-days

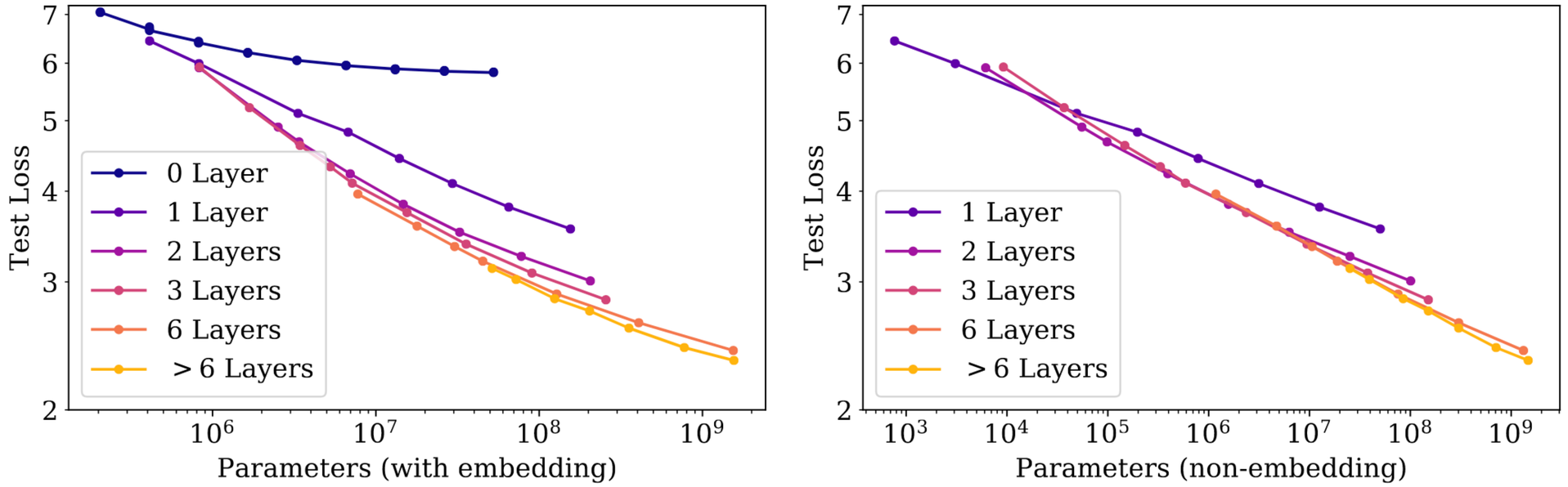
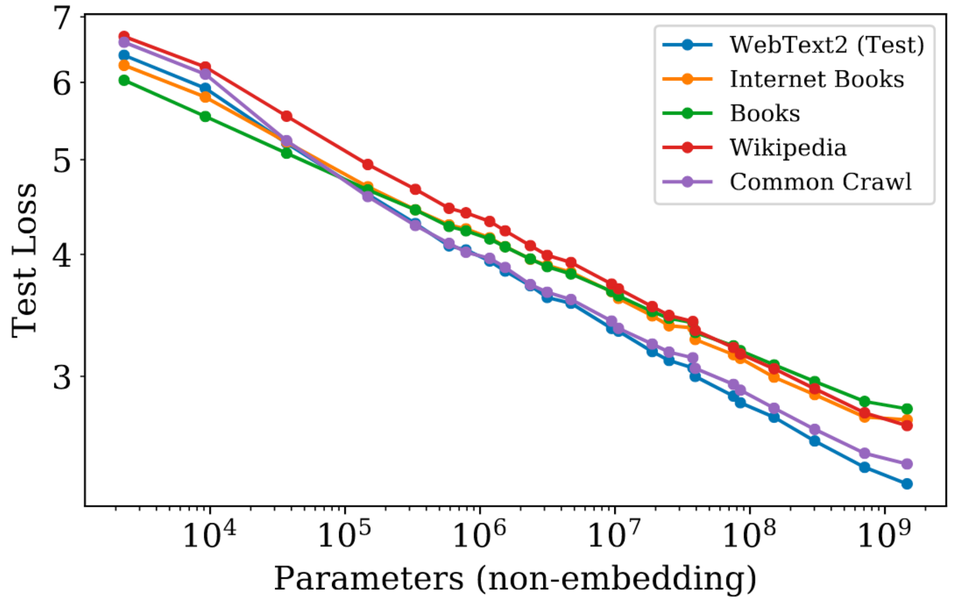


Figure 6 **Left:** When we include embedding parameters, performance appears to depend strongly on the number of layers in addition to the number of parameters. **Right:** When we exclude embedding parameters, the performance of models with different depths converge to a single trend. Only models with fewer than 2 layers or with extreme depth-to-width ratios deviate significantly from the trend.

- "If we instead use the total parameter count (including the embedding parameters) the trend is somewhat obscured."
- "This suggests that the embedding matrix can be made smaller without impacting performance."



- Figure 8. Left.
- Although these models have been **trained on the WebText2 dataset,** their **test loss on a variety of other datasets** is also a power-law in N with nearly identical power, as shown in Figure 8.

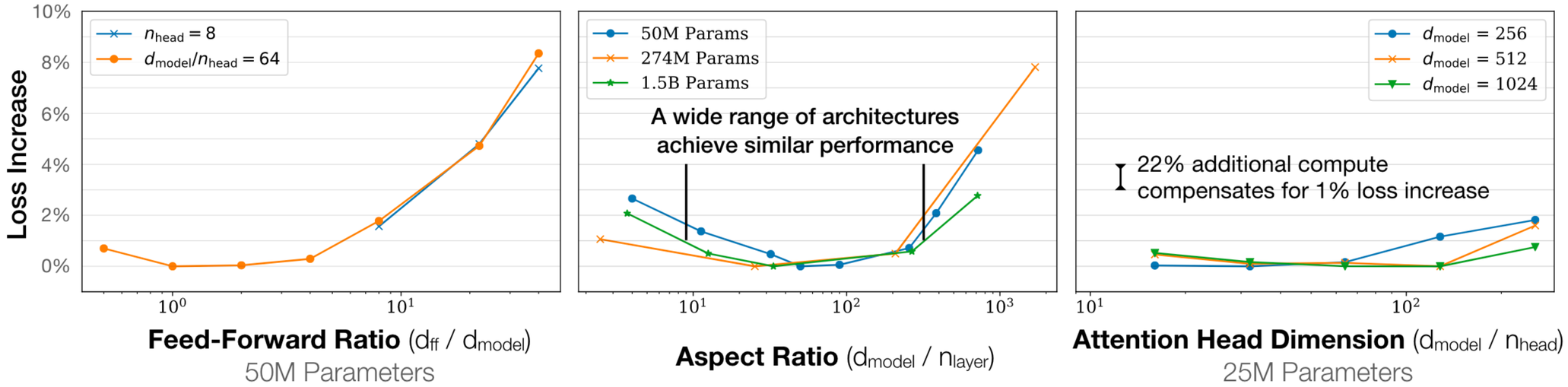


Figure 5 Performance depends very mildly on model shape when the total number of non-embedding parameters N is held fixed. The loss varies only a few percent over a wide range of shapes. Small differences in parameter counts are compensated for by using the fit to $L(N)$ as a baseline. Aspect ratio in particular can vary by a factor of 40 while only slightly impacting performance; an $(n_{\text{layer}}, d_{\text{model}}) = (6, 4288)$ reaches a loss within 3% of the $(48, 1600)$ model used in [RWC+19].

- N is fixed.
- d means dimension. n means number. ff means feed-forward layer.

Conclusion 3

Universality of Overfitting

- “Performance improves predictably as long as we scale up N and D **in tandem.**”
- “Every time we increase the model size 8x, we only need to increase the data by roughly 5x **to avoid a penalty.**”

4.1 Proposed $L(N, D)$ Equation

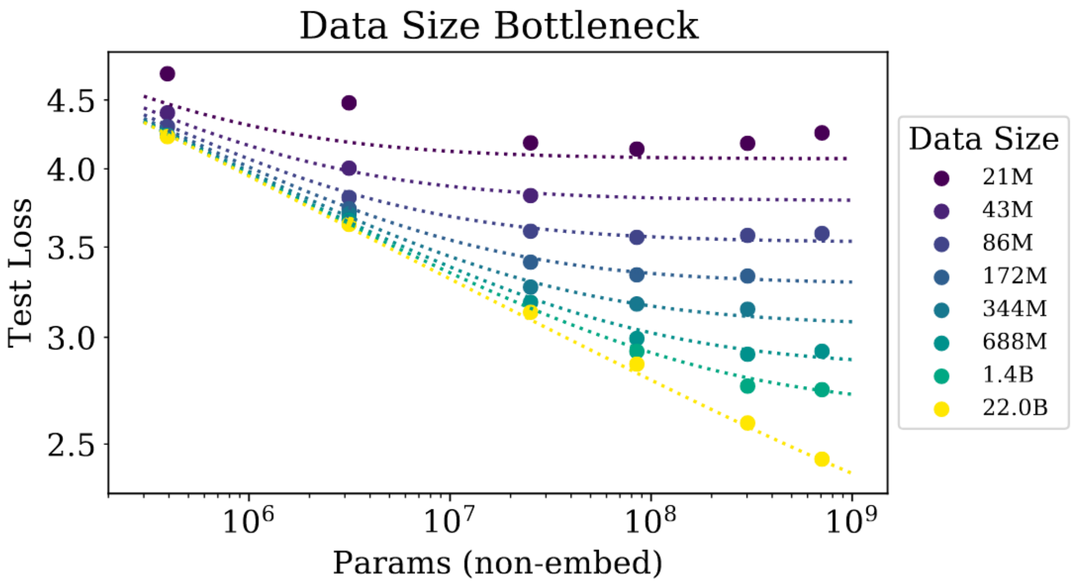
We have chosen the parameterization (1.5) (repeated here for convenience):

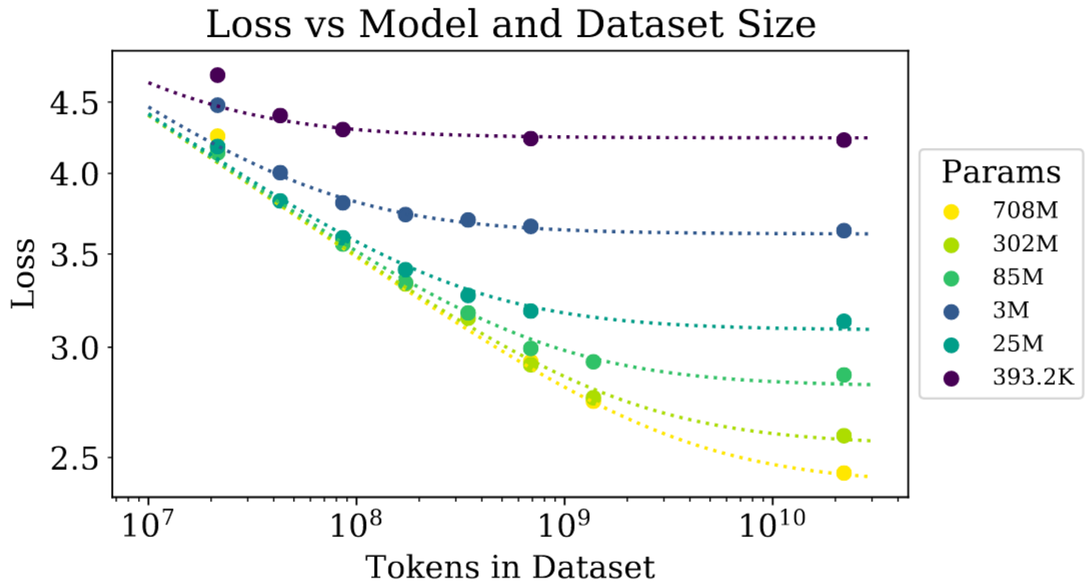
$$L(N, D) = \left[\left(\frac{N_c}{N} \right)^{\frac{\alpha_N}{\alpha_D}} + \frac{D_c}{D} \right]^{\alpha_D} \tag{4.1}$$

using three principles:

1. Changes in vocabulary size or tokenization are expected to rescale the loss by an overall factor. The parameterization of $L(N, D)$ (and all models of the loss) must naturally allow for such a rescaling.
2. Fixing D and sending $N \to \infty$, the overall loss should approach $L(D)$. Conversely, fixing N and sending $D \to \infty$ the loss must approach $L(N)$.
3. $L(N, D)$ should be analytic at $D = \infty$, so that it has a series expansion in $1/D$ with integer powers. Theoretical support for this principle is significantly weaker than for the first two.

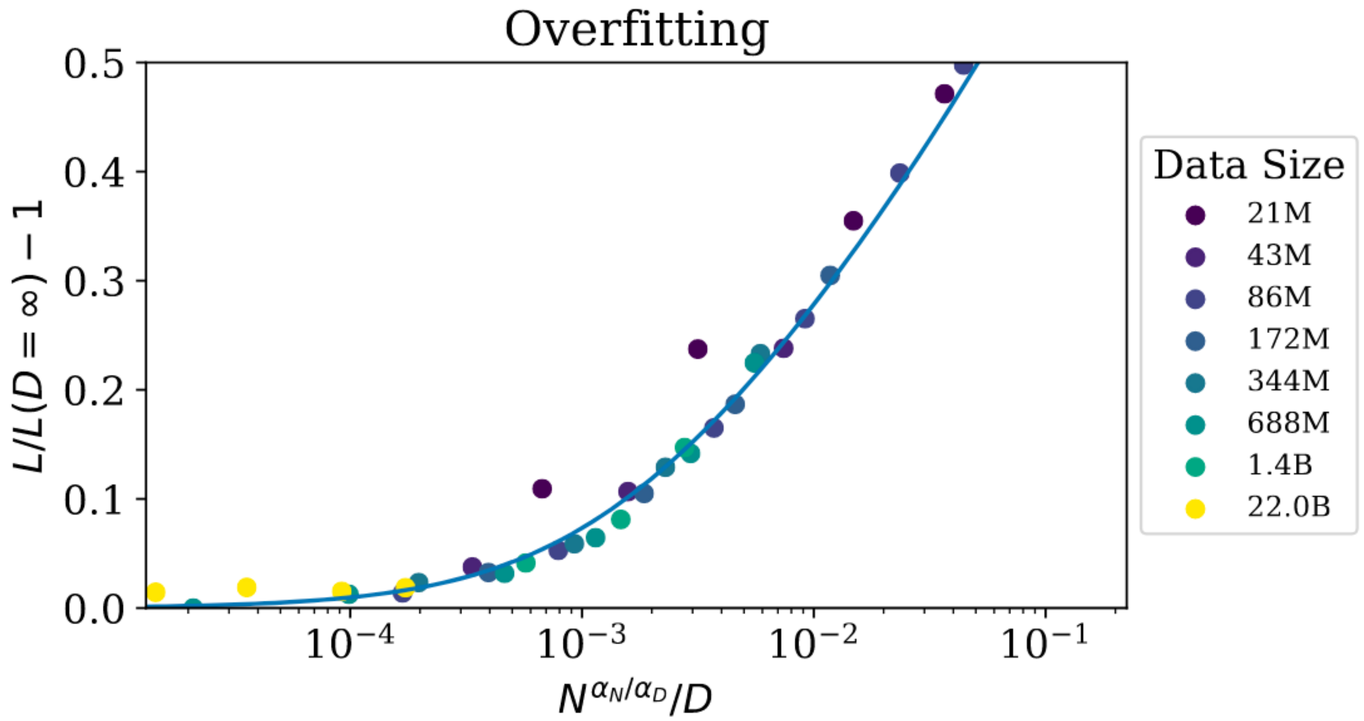
Our choice of $L(N, D)$ satisfies the first requirement because we can rescale N_c, D_c with changes in the vocabulary. This also implies that the values of N_c, D_c have no fundamental meaning.





- Figure 9. Left
- For large D, performance is a straight power law in N.
- For a smaller fixed D, performance stops improving as N increases and the model begins to **overfit**.

- Figure 4. Left



- The extent of overfitting depends predominantly on the ratio $N^{\alpha_N/\alpha_D}/D$, as predicted in equation (4.3). The line is our fit to that equation.

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

Overfitting

We estimate that the variation in the loss with different random seeds is roughly 0.02, which means that to avoid overfitting when training to within that threshold of convergence we require

$$D \gtrsim (5 \times 10^3)\, N^{0.74} \tag{4.4}$$

- Empirical inequation.
- Compare the variation in the loss across seeds because we do not want the model to overfit a certain seed.

Conclusion 4

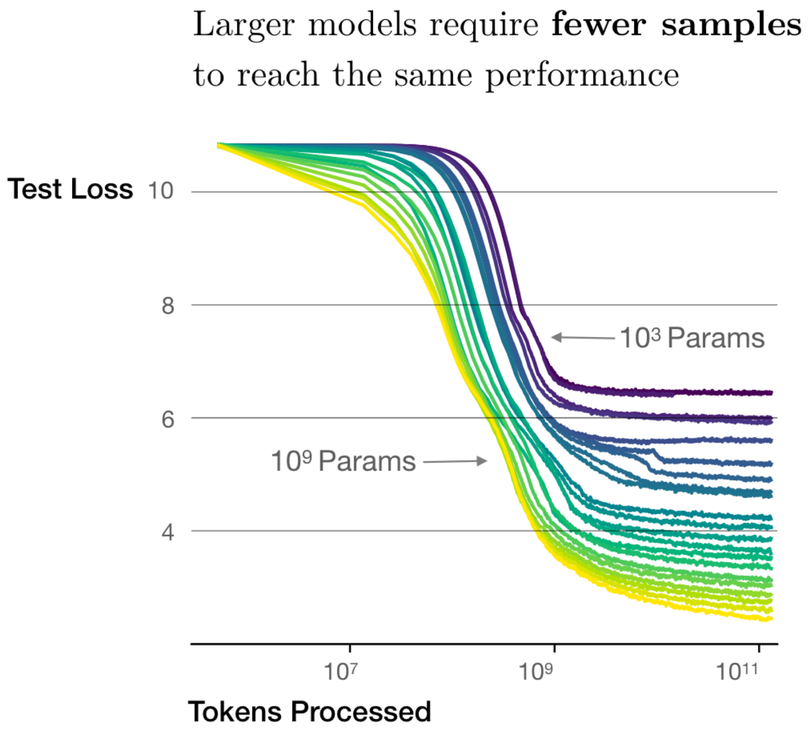
Universality of Training

- **Training curves** follow predictable power-laws whose parameters are roughly **independent** of the model size.
- By extrapolating the early part of a training curve, we can roughly predict the loss that would be achieved if we trained for much longer. (Section 5)
- (I skip the details here.)

Conclusion 5

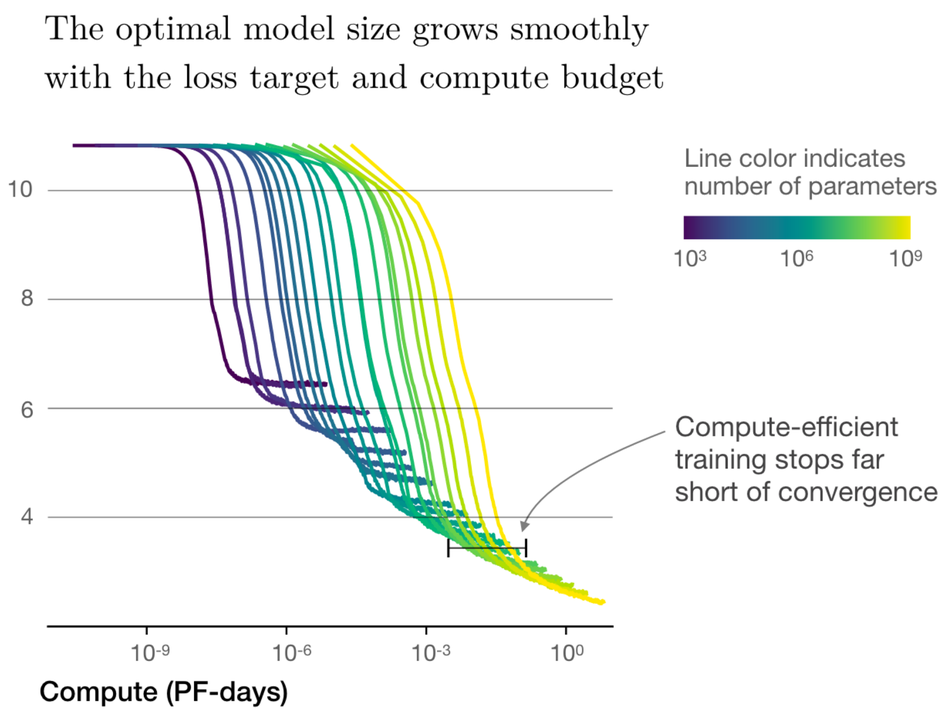
Sample Efficiency

- Large models are more sample-efficient than small models, reaching the same level of performance with fewer optimization steps (Figure 2).



Conclusion 6

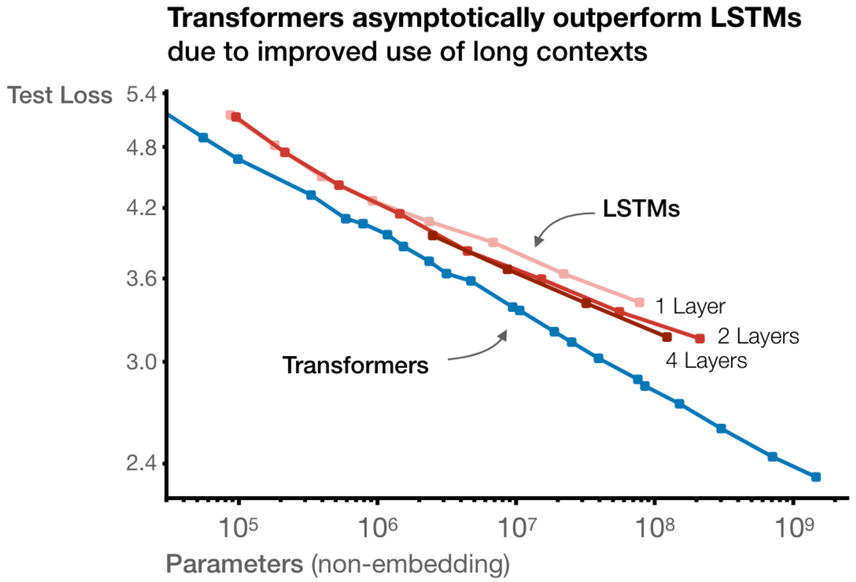
Convergence is Inefficient



- When working with a fixed computational budget C, and there are no other restrictions on the model size N or available data D,
    - we can achieve optimal performance by “training very large models and stopping shortly before convergence.”
    - “This approach is far more sample-efficient than the expectation of 'training small models to convergence.”

Marginal Conclusion 1

Transformer outperforms LSTM



- Section 3.2.1. Figure 7. Left.
- GPT is used for NLG.
  - So it has to use its own output as input. Does this mean that the time complexity is high, like RNNs? Doesn’t this negate the advantage of attention?
  - Complexity per layer is indeed worse than Transformer, but “RNNs struggle with long sequences mostly because of poor dependency distance,” the advantage of attention's max path length still exists.

GPT-3

OpenAI. 2020.5. “Language Models are Few-Shot Learners.”

Tom B. Brown* **Benjamin Mann*** **Nick Ryder*** **Melanie Subbiah***

Jared Kaplan† **Prafulla Dhariwal** **Arvind Neelakantan** **Pranav Shyam**

Girish Sastry **Amanda Askell** **Sandhini Agarwal** **Ariel Herbert-Voss**

Gretchen Krueger **Tom Henighan** **Rewon Child** **Aditya Ramesh**

Daniel M. Ziegler **Jeffrey Wu** **Clemens Winter**

Christopher Hesse **Mark Chen** **Eric Sigler** **Mateusz Litwin** **Scott Gray**

Benjamin Chess **Jack Clark** **Christopher Berner**

Sam McCandlish **Alec Radford** **Ilya Sutskever** **Dario Amodei**

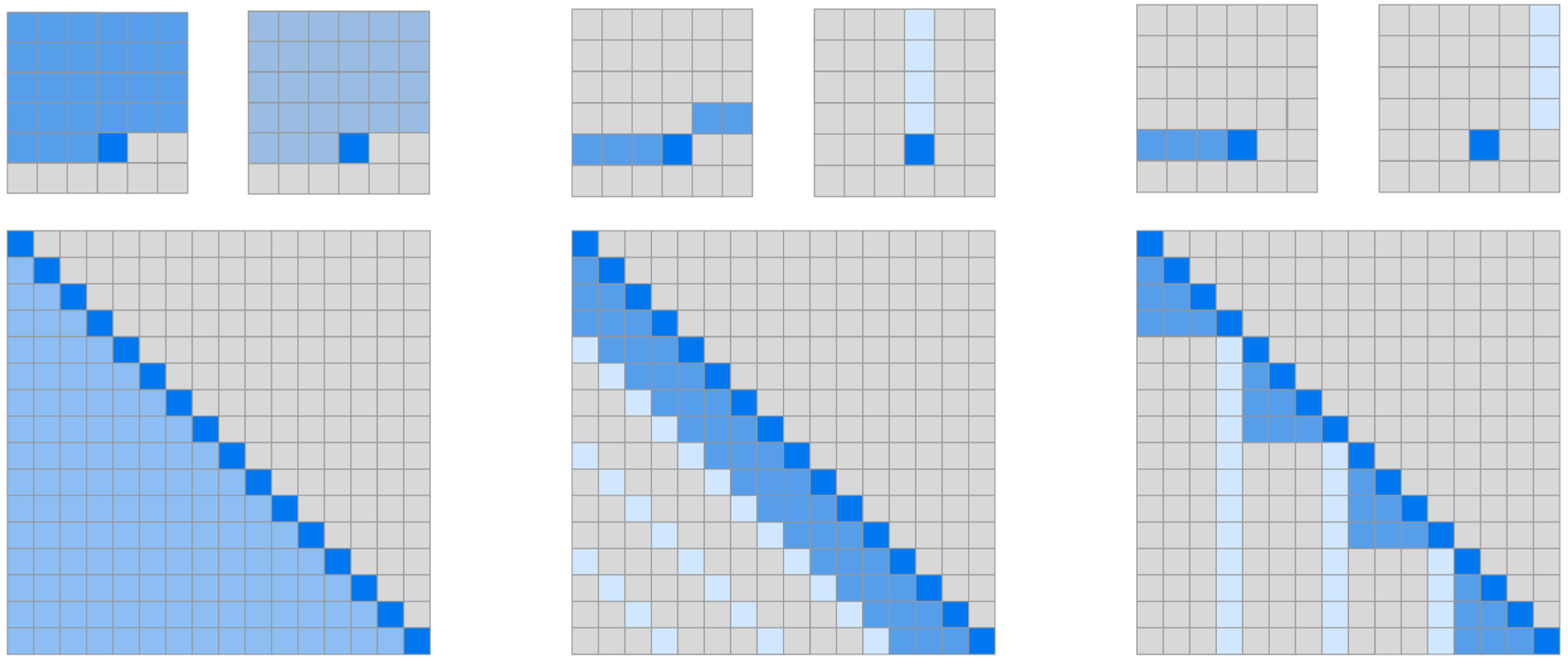
GPT-3

- GPT-3 = GPT-2 + More Parameters + Larger Dataset + Sparse Attention + Few-Shot Learning
- (The paper is more like a technical report.)

Sparse Attention

- GPT-3 = GPT-2 + More Parameters + Larger Dataset + Sparse Attention + Few-Shot Learning
    - Sparse attention
        - Full attention: Calculate the similarity of the current token with all other tokens. (Or with the token ahead of it, if it is masked.)
        - It is more efficient than full attention.

Sparse Attention Example



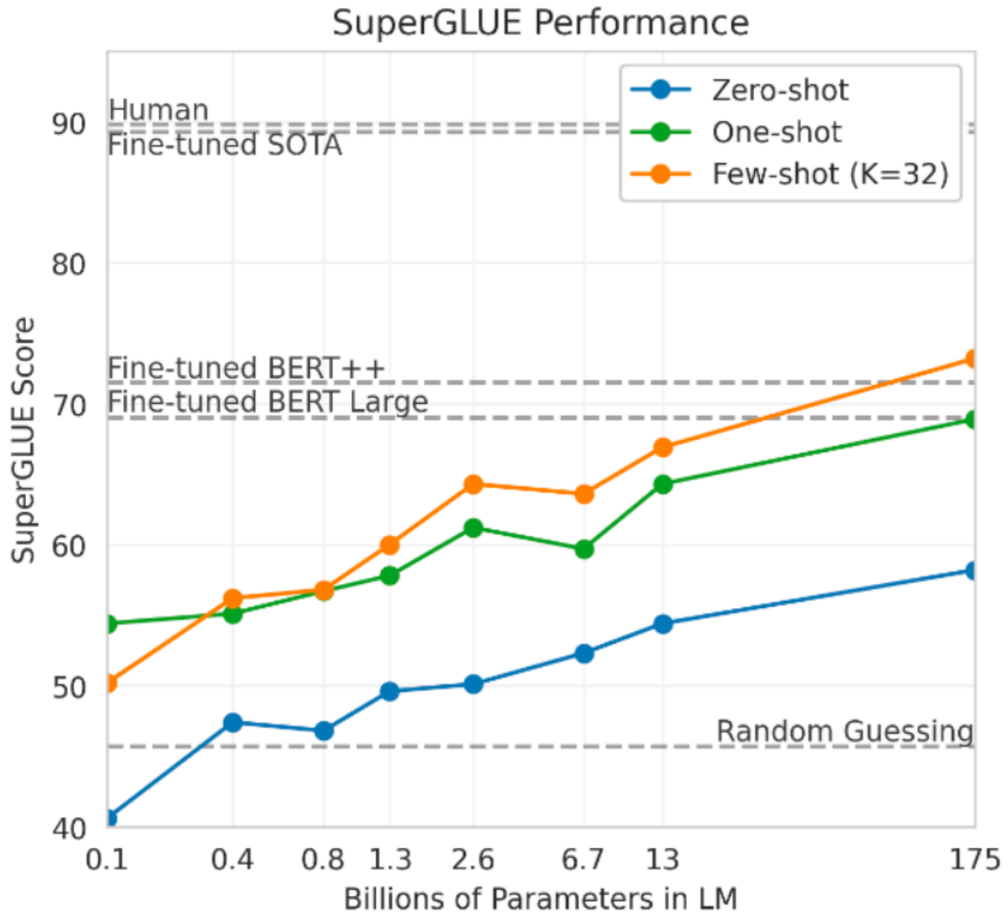
(a) Transformer (b) Sparse Transformer (strided) (c) Sparse Transformer (fixed)

- Figure 3 in “Generating Long Sequences with Sparse Transformers.” This paper proposed sparse attention. And this example is for a 2D transformer.
- The sparse attention details in GPT-3 are not discussed in the paper.

Few-Shot Learning

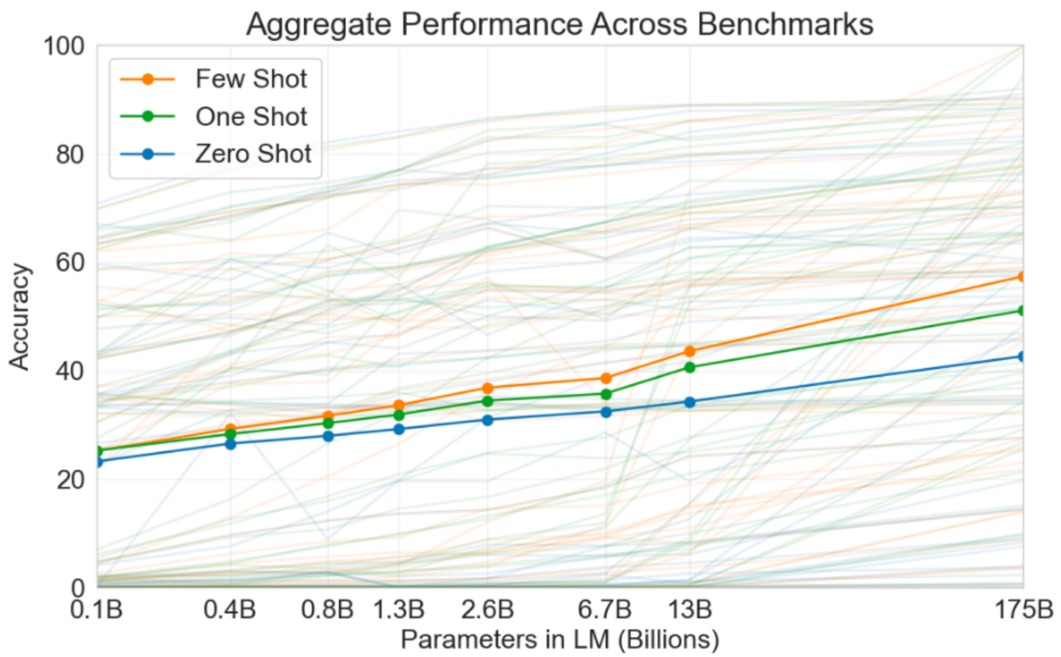
- GPT-1: Pre-Training + Fine-Tuning. The model will **be updated** for new in fine-tuning phase for specific tasks. Fine-tuning can make the model perform better in these tasks.
- GPT-2: Pre-Training. No fine-tuning. It sees a lot of data and several specific tasks are reconstructed to be used in pre-training phase. It is also called **zero-shot learning.**
- GPT-3: Pre-Training + Few-shot Learning. The user will feed it some examples of the questions first and then ask it the true question. No update of the model is needed.
    - The model can understand the task because the output is generated depending on the input, which follows the NLG pattern of “predicting the next word.”
    - In-context learning. Prompting.

Scaling Law



- Human and fine-tuned SOTA are far better than others.
- Few-shots > one-shots > zero-shot
- Scaling law

Scaling Law



- Human and fine-tuned SOTA are far better than others.
- Few-shots > one-shots > zero-shot
- Scaling law

Saturated Examples