Scaling Laws for Neural Language Models

OpenAI: [KMH+20]

Александра Муравьёва,
172 группа

11 ноября 2020

Intro

Goal of study

What empirical laws can we draw from training Transformers varying a wide range of hyperparameters?

Hyperparameters:

- Model size (ranging in size from 768 to 1.5 billion non-embedding parameters)
- Dataset size (ranging from 22 million to 23 billion tokens)
- Shape (including depth, width, attention heads, and feed-forward dimension)
- Context length (1024 for most runs, though they also experiment with shorter contexts)
- Batch size (2^{19} for most runs, but they also vary it to measure the critical batch size)
- Compute budget

Experiment design

Training corpus: WebText2

Models: Transformers of different shape and size, LSTM

Optimizing: autoregressive cross-entropy loss averaged over a 1024-token context

Training procedure:

- Adam optimizer for a fixed number of steps with a batch size of 512 sequences of 1024 tokens
- Adafactor for large models (>1B parameters)
- fixed learning rate schedule (because it was found that the results at convergence are independent of lr)

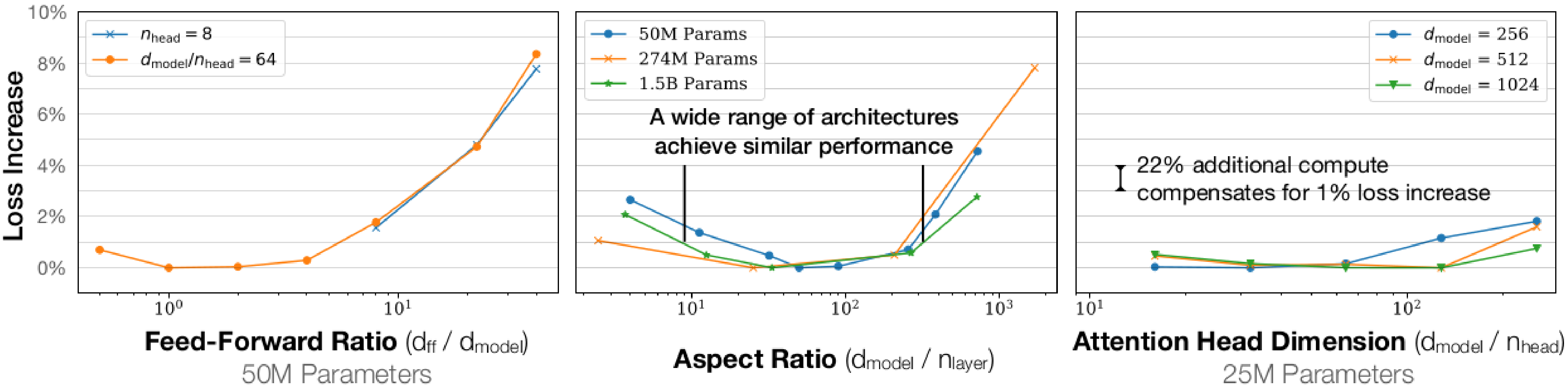
Notation

- L – cross entropy loss in nats
- N – number of model parameters, excluding all vocabulary and positional embeddings
- $C \approx 6NBS$ – an estimate of the total non-embedding training compute, where B is the batch size, and S is the number of training steps
- D – dataset size in tokens
- n_{layer} – number of layers
- d_{model} – dimension of the residual stream
- d_{ff} – dimension of the intermediate feed-forward layer
- d_{attn} – dimension of the attention output
- n_{heads} – number of attention heads per layer

Key findings

1. Performance depends **strongly** on:
   - N – number of parameters
   - D – dataset size
   - C – amount of compute,

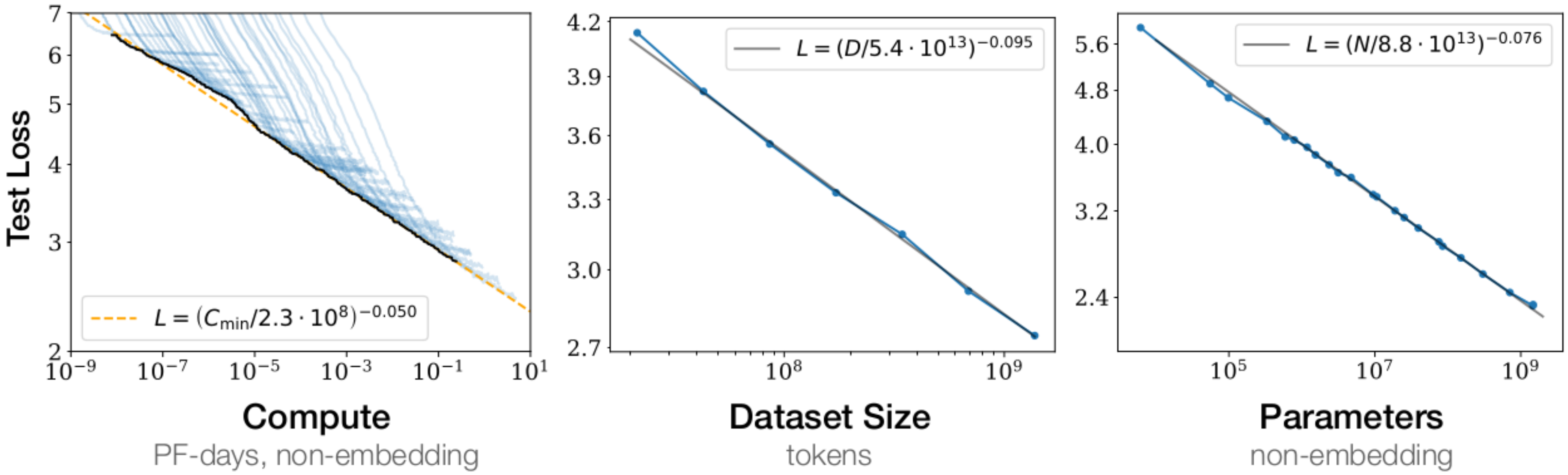
   weakly – on:
   - architecture, e.g. width, depth, number of self-attention heads



1. Performance depends **strongly** on:
   - N – number of parameters
   - D – dataset size
   - C – amount of compute,

   weakly – on:
   - architecture, e.g. width, depth, number of self-attention heads

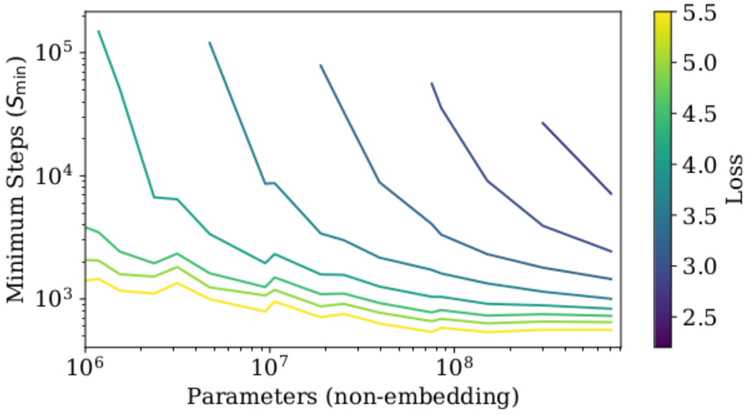
2. Performance has a power-law relationship with each of N, D, C

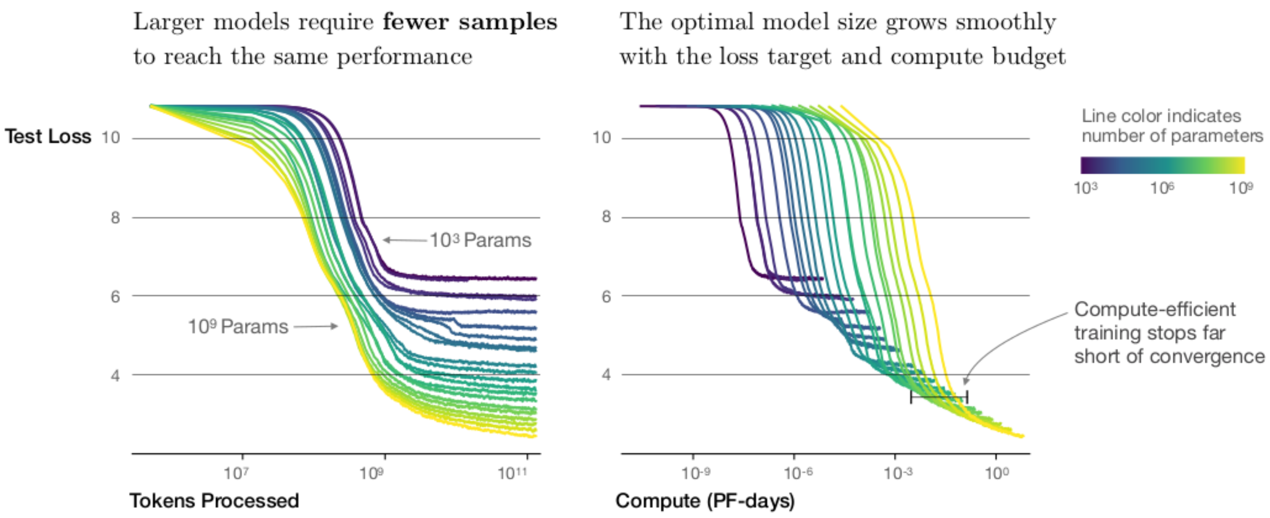


3. Sample and compute efficiency:

large models are more sample-efficient than small models

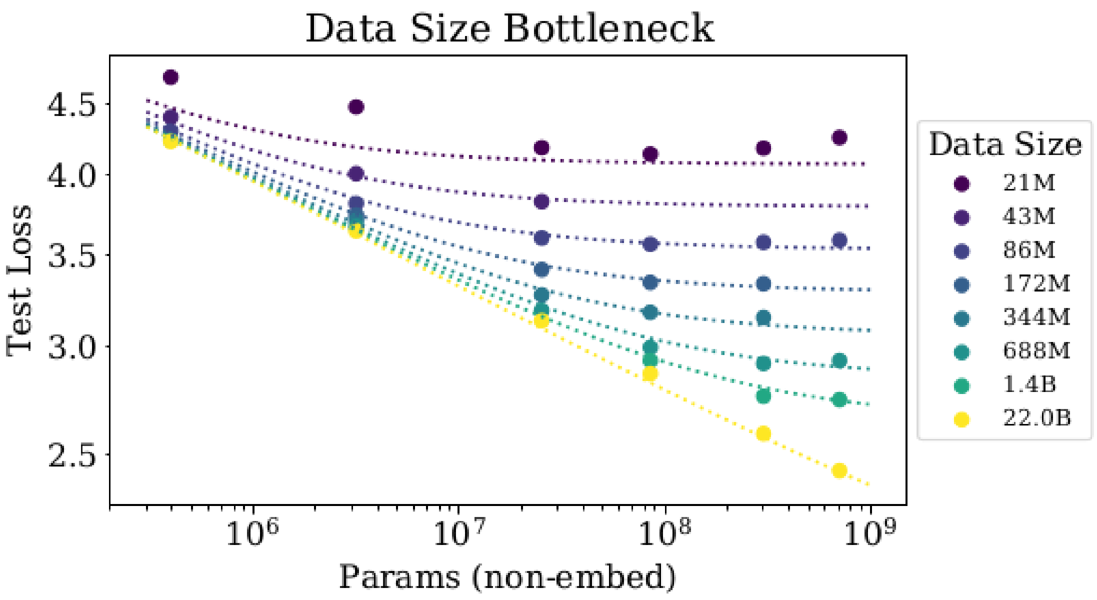
→ use **large** models with **early stopping**!





4. Overfitting rule:

with every **8x model size** increase use **5x data size** increase to avoid overfitting



How to get this rule

Parametrization

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

$$L(N) \approx \left(\frac{N_c}{N}\right)^{\alpha_N}$$

$$L(D) \approx \left(\frac{D_c}{D}\right)^{\alpha_D}$$

Parameter	α_N	α_D	N_c	D_c
Value	0.076	0.103	6.4×10^{13}	1.8×10^{13}

Table 2 Fits to $L(N, D)$

$$\delta L(N, D) \equiv \frac{L(N, D)}{L(N, \infty)} - 1 \Rightarrow \delta L \approx \left(1 + \left(\frac{N}{N_c}\right)^{\frac{\alpha_N}{\alpha_D}} \frac{D_c}{D}\right)^{\alpha_D} - 1$$

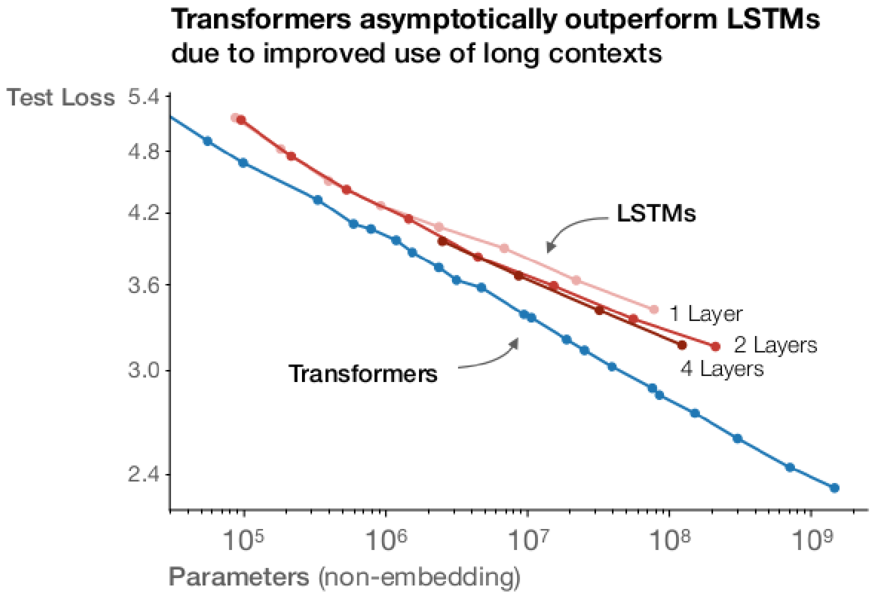
should be less than loss variation, which is roughly 0.02

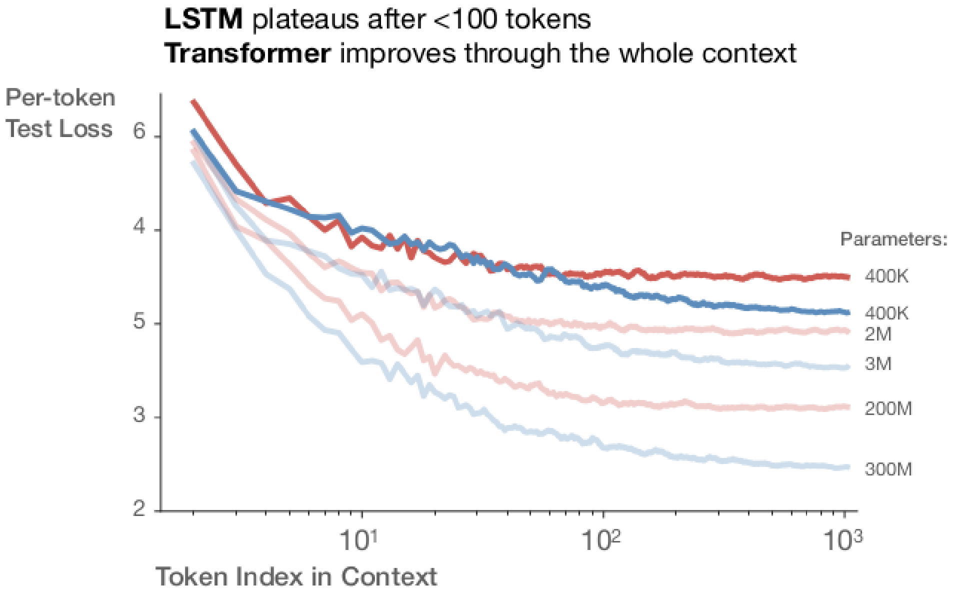
$$\Rightarrow \quad D \gtrsim (5 \times 10^3)\, N^{0.74} \quad \Rightarrow$$

with **8x model size** increase
use **5x data size** increase

5. Comparing to LSTMs

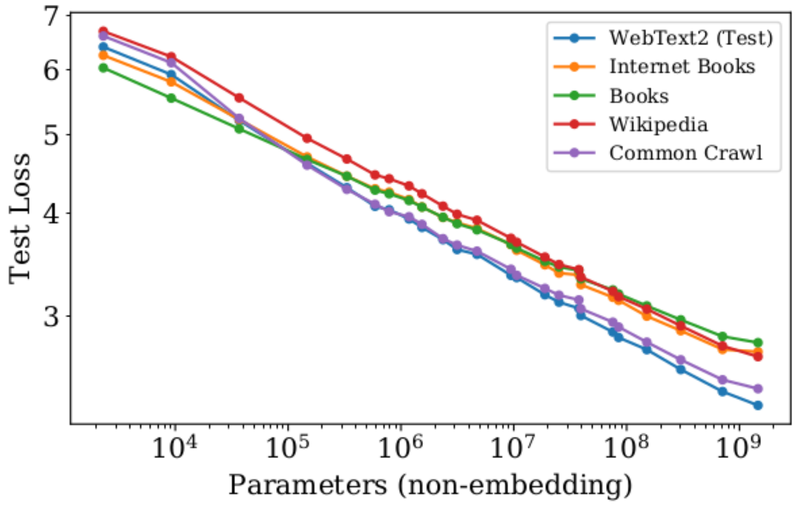
- perform equally well for tokens early in context
- overall perfomance is worse due to later tokens

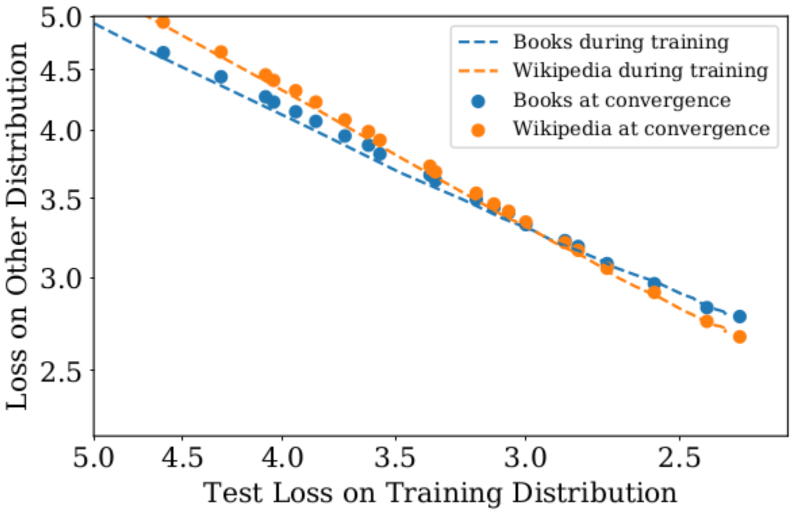




6. Generalization among data distributions

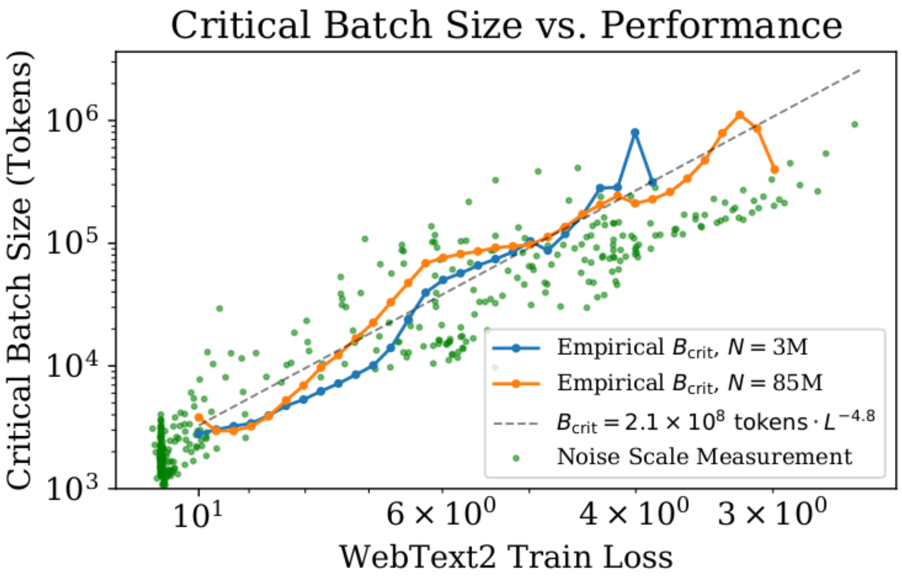
performance depends only on **training distribution performance** and has a nearly **constant offset**





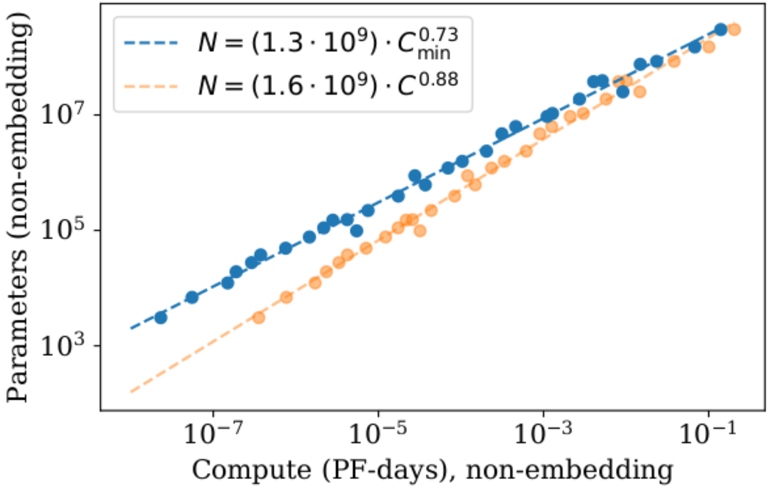
7. Critical batch size:

- Training time / compute balance: train at batch size B ≈ B_{crit}
- B > B_{crit}: degradation in compute-efficiency
- B_{crit} does not depend on model size and also has a power-law $$B_{\text{crit}}(L) \approx \frac{B_*}{L^{1/\alpha_B}}$$
- ~1-2 million tokens for largest models



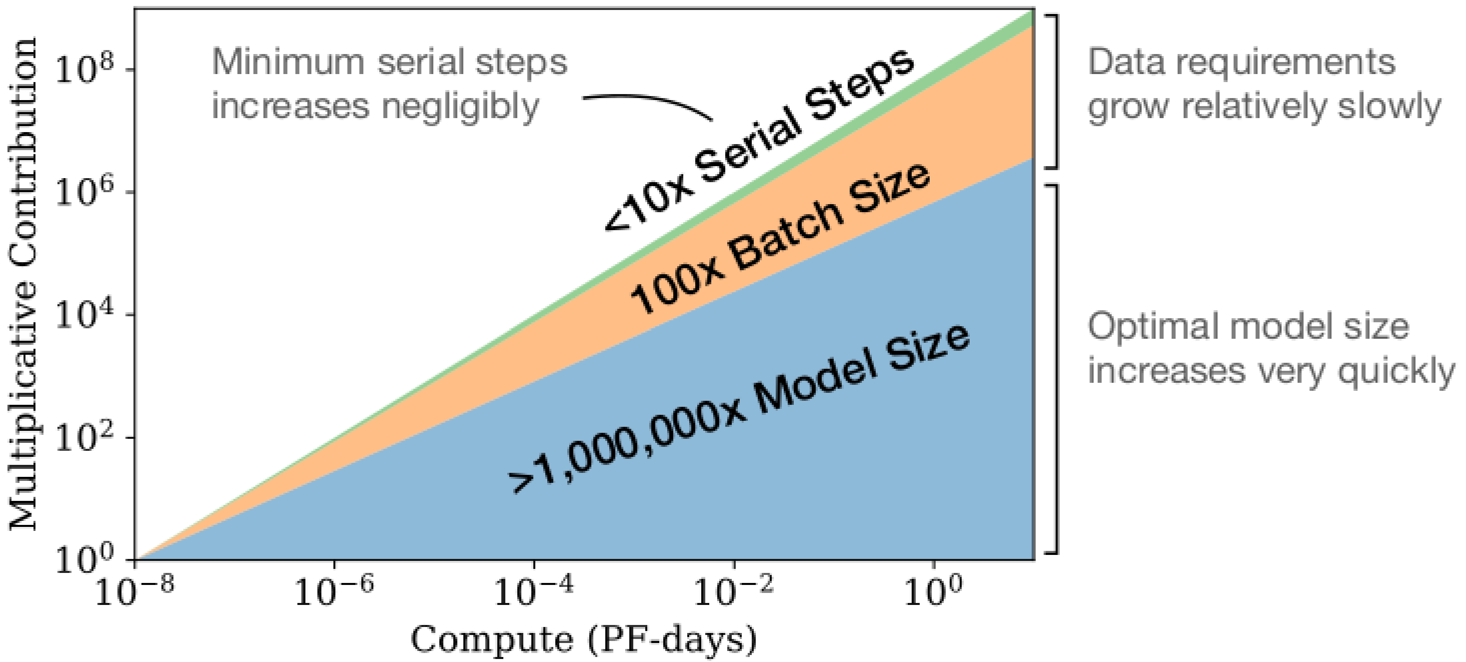
8. Optimal allocation of a large compute budget (C_{min})

- choose B == B_{crit}
- fit the power law $N(C_{\min}) \propto (C_{\min})^{\alpha_C^{\min}/\alpha_N}$ where $\alpha_C^{\min} \equiv \frac{1}{1/\alpha_S + 1/\alpha_B + 1/\alpha_N}$ on small models and choose the optimal N
- D should be chosen so as not to overfit: $D \propto N^{0.74}$
- By definition $C_{\min} \equiv 6NB_{\text{crit}}S$, which gives S – number of training steps



9. Relative importance of N, B, S when increasing compute

- predominantly increase the model size N
- simultaneously scale up the batch size via $B \propto B_{crit}$
- number of steps S will increase negligibly



Вопросы

1. Какие тренды были замечены авторами статьи в обучении трансформеров?
2. Уравнение зависимости лосса от числа параметров модели и количества данных L(N, D) и зачем оно нужно.
3. Что такое критический размер батча?
4. Как максимизировать эффективность фиксированного compute budget?

References

[KMH+20]

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.