Week 4 Assignment

Generative AI

Saarland University – Winter Semester 2024/25

Martínez
7057573
cama00005@stud.uni-saarland.de

1 Exercise Assignment: E1

Given:

- $N = 67$ billion parameters
- $D = 4.1$ trillion training tokens
- Constants for loss estimation: $E = 1.69; A = 406.4; B = 410.7; \alpha = 0.34; \beta = 0.28$

The approximate compute cost C (measured in FLOPs) required to train the model for the given N and D can be approximated as [1]:

$$C = \text{FLOPs}(N, D) \approx 6ND$$

Thus,

$$C \approx 6 \times 67 \times 10^9 \times 4.1 \times 10^{12} = 1.65 \times 10^{24} \text{ FLOPs}$$

And the expected loss L is given by:

$$L(N, D) = E + \frac{A}{N^\alpha} + \frac{B}{D^\beta} = 1.69 + \frac{406.4}{(67 \times 10^9)^{0.34}} + \frac{410.7}{(4.1 \times 10^{12})^{0.28}} \approx 1.90$$

2 Exercise Assignment: E2

To improve the model's performance decreasing the previously computed loss $L = 1.90$ by 1%, we would need a new loss L' such that:

$$L' = 0.99 \times L = 0.99 \times 1.90 \approx 1.88$$

Now, we can calculate the new amount of training tokens D' required to achieve this new loss L' assuming the number of parameters N remains the same:

$$L' = E + \frac{A}{N^\alpha} + \frac{B}{D'^\beta} \to D' = \left(\frac{B}{L' - E - \frac{A}{N^\alpha}} \right)^{\frac{1}{\beta}} = \left(\frac{410.7}{1.88 - 1.69 - \frac{406.4}{(67 \times 10^9)^{0.34}}} \right)^{\frac{1}{0.28}}$$
$$\approx 6.48 \text{ trillion tokens}$$

The new total number of FLOPs C' required for this training would then be:

$$C' \approx 6ND' = 6 \times 67 \times 10^9 \times 6.48 \times 10^{12} = 2.6 \times 10^{24} \text{ FLOPs}$$

Finally, with the following geometric series, we can estimate the number of additional training epochs x after the first epoch required to achieve the new loss L':

$$D' = 2 \cdot \left(1 - \frac{1}{2^{x+1}} \right) \cdot D \to x = \log_2 \left(\frac{1}{1 - \frac{D'}{2D}} \right) \tag{1}$$

Replacing the corresponding values in (1), we get:

$$x = \log_2 \left(\frac{1}{1 - \frac{6.48}{2 \times 4.1}} \right) \approx 1.25 \text{ epochs}$$

3 Exercise Assignment: E3

For compute optimal training, the number of epochs x required to train the model with N' parameters is given by:

$$C' = 6N'D \rightarrow D = \frac{C'}{6 \times 1.41N} = \frac{2.6 \times 10^{24}}{6 \times 1.41 \times 67 \times 10^9} \approx 4.59 \text{ trillion tokens}$$

Now, using (1) with the new number of training tokens D, we can calculate the optimal number of epochs x:

$$x = \log_2 \left(\frac{1}{1 - \frac{4.59}{2 \times 4.1}} \right) \approx 1.18 \text{ epochs}$$

TODO: For the above scenario where the available training data is constrained, what conclusions can you draw about the optimal scaling of model size and number of epochs w.r.t. the compute cost?

4 Exercise Assignment: E4

GPT-3 has 175 billion parameters, each with 16-bit (2 bytes) precision. This means that one single GPT-3 model would occupy in memory[1]:

$$175 \times 10^9 \text{ parameters} \times 2 \text{ bytes/parameters} = 3.50 \times 10^{11} \text{ bytes} \quad (\approx 325.96 \text{ GB})$$

On the other hand, 100 distinct tasks require finetuning 100 different models. This means that the total amount of memory required to store all these models is:

$$100 \times 3.50 \times 10^{11} \text{ bytes} = 3.50 \times 10^{13} \text{ bytes} \quad (\approx 31.83 \text{ TB})$$

5 Exercise Assignment: E5

If instead of full-finetuning, we use adapters to fine-tune only the query and value projection matrices in the self-attention module, we would need:

$$\begin{aligned} 2 \times d_{\text{model}} \times d_{\text{model}} \times 96 \times 2 \text{ bytes} &= 2 \times 12{,}288 \times 12{,}288 \times 96 \times 2 \text{ bytes/parameters} \\ &\approx 6.04 \times 10^8 \text{ bytes} \quad (54 \text{ GB}) \end{aligned}$$

Since we have 2 matrices (query and value matrices) of size $d_{\text{model}} \times d_{\text{model}}$ per layer, where $d_{\text{model}} =$ 12,288 for GPT-3, and 96 layers.

6 Exercise Assignment: E6

If we apply low-rank adaptation (LoRA) with rank $r = 4$ to the query $Q \in \mathbb{R}^{d_{\text{model}} \times d_{\text{model}}}$ and value $V \in \mathbb{R}^{d_{\text{model}} \times d_{\text{model}}}$ projection matrices in each of the 96 layers, we need to define 96 trainable matrices $B \in \mathbb{R}^{d_{\text{model}} \times 4}$ and $A \in \mathbb{R}^{4 \times d_{\text{model}}}$. Thus, in this setting, we would need:

$$\begin{aligned} 2 \times d_{\text{model}} \times 4 \times 96 \times 2 \text{ bytes} &= 2 \times 12{,}288 \times 4 \times 96 \times 2 \text{ bytes} \\ &\approx 1.89 \times 10^7 \text{ bytes} \quad (18 \text{ MB}) \end{aligned}$$

[1] 1 GB $= 1024^3$ bytes

7 Exercise Assignment: E7

The value of rank r for LoRA if we adapt only the query $Q \in \mathbb{R}^{d_{\text{model}} \times d_{\text{model}}}$ projection matrix this time, and we restrict ourselves with the same amount of memory as in (E.6), would have to be:

$$M = d_{\text{model}} \times r \times 96 \times 2 \text{ bytes} \rightarrow r = \frac{M}{d_{\text{model}} \times 96 \times 2 \text{ bytes}} = \frac{1.89 \times 10^7}{12{,}288 \times 96 \times 2} = 8$$

8 Individual Assignment: I3

9 Individual Assignment: I7

10 Individual Assignment: I8

Acknowledgements

This week's slides.

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. *CoRR*, abs/2001.08361, 2020.