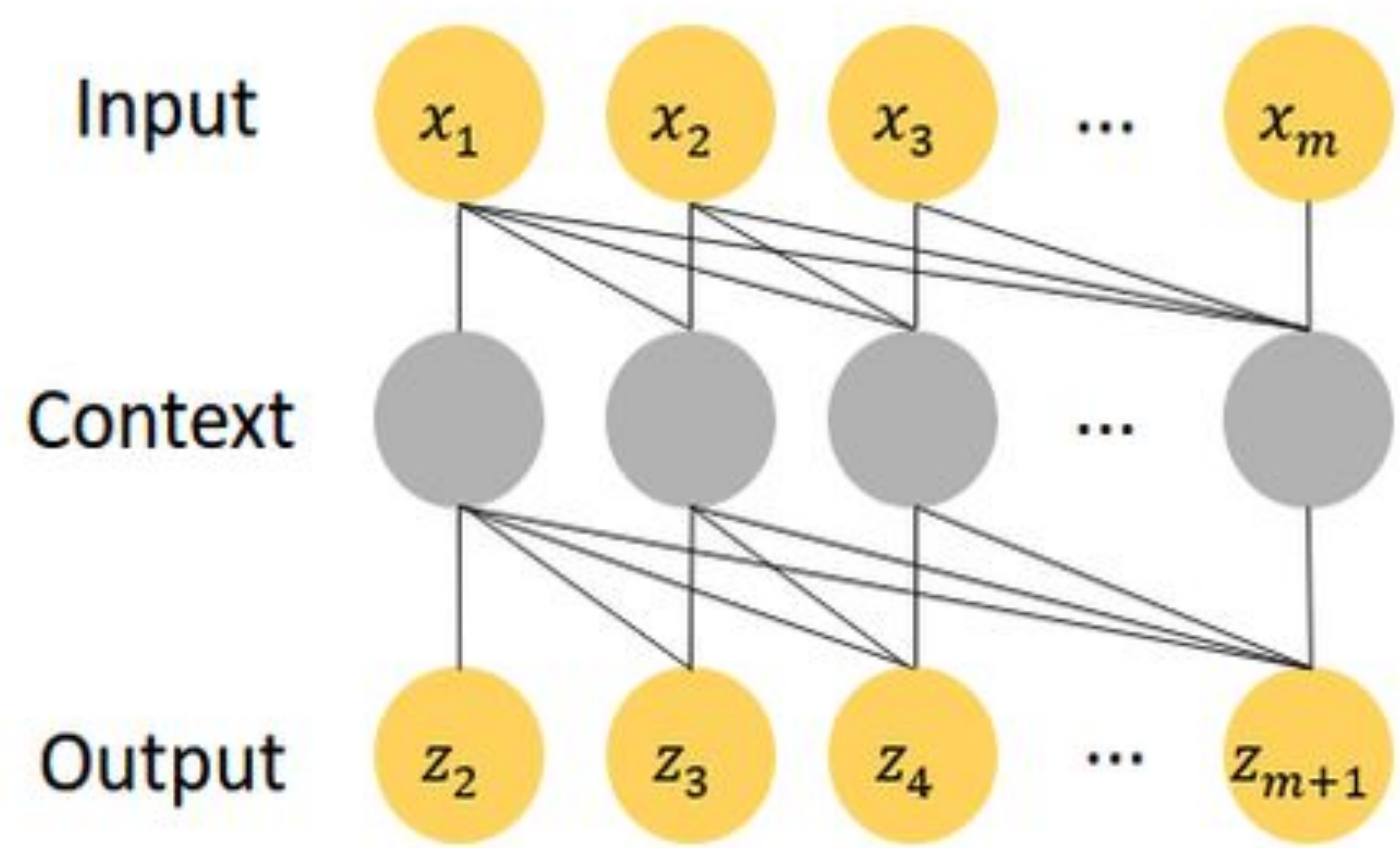
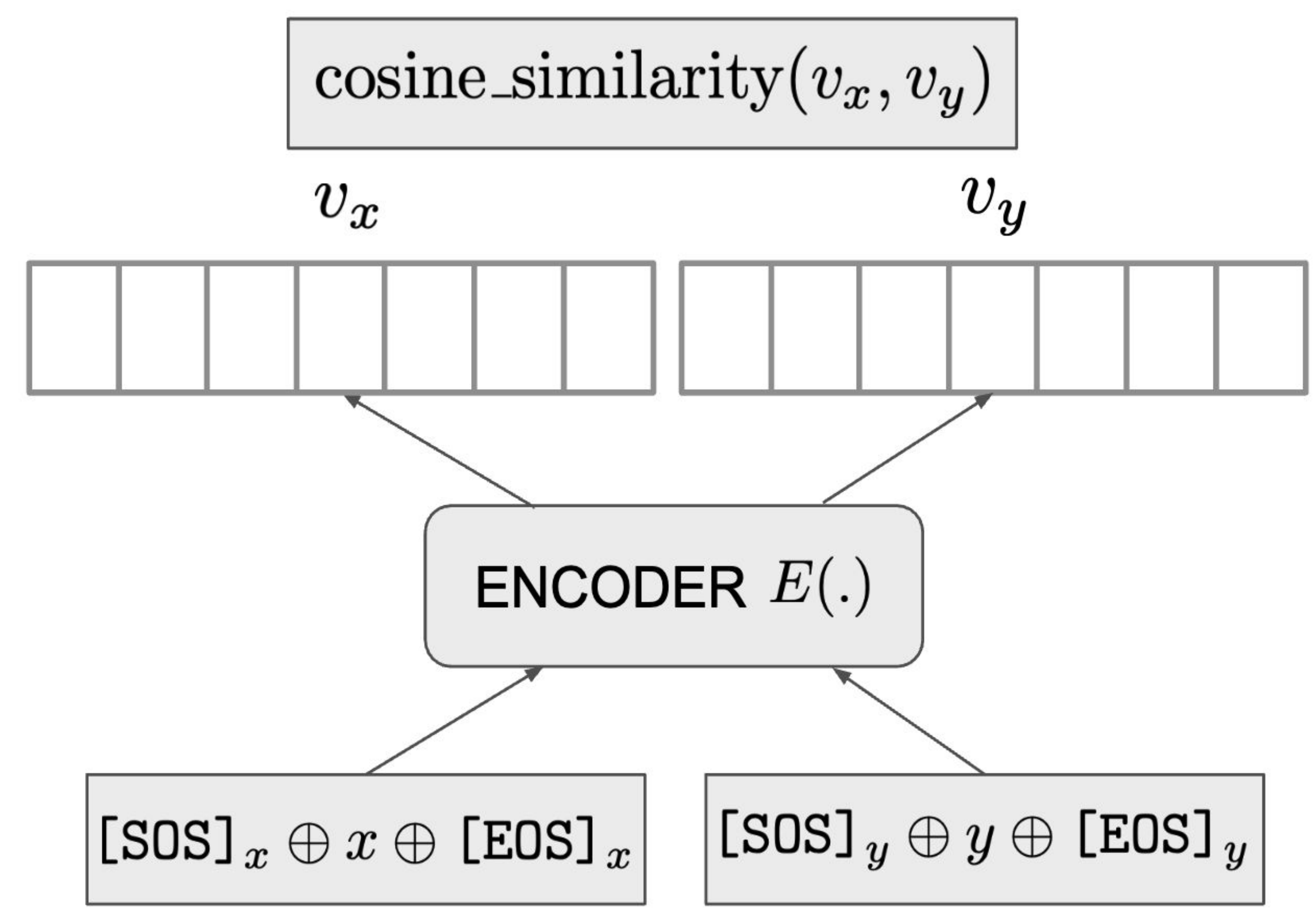
Repurposing Language Models into Embedding Models: Finding the Compute-Optimal Recipe

Albert Q. Jiang*, Alicja Ziarko*, Bartosz Piotrowski, Wenda Li, Mateja Jamnik†, Piotr Miłoś†
*: equal leading contributions. †: equal advising contributions.

Decoders to encoders via contrastive learning





Given a fixed computational budget:

- How large a model should I use?
- How much data do I need?
- Are PEFT methods better than full fine-tuning? What hyperparameters should I choose?

Optimising loss given fine-tuning budget

We choose the Pythia family of pre-trained decoder models and experiment with computational budgets from 1.5e15 to 1.5e18 FLOP by varying

- Model parameters
- Amount of data
- Parameter-Efficient Fine-Tuning (PEFT) methods
  - Block freezing
  - LoRA
  - Bias tuning

Cost

Data quantity

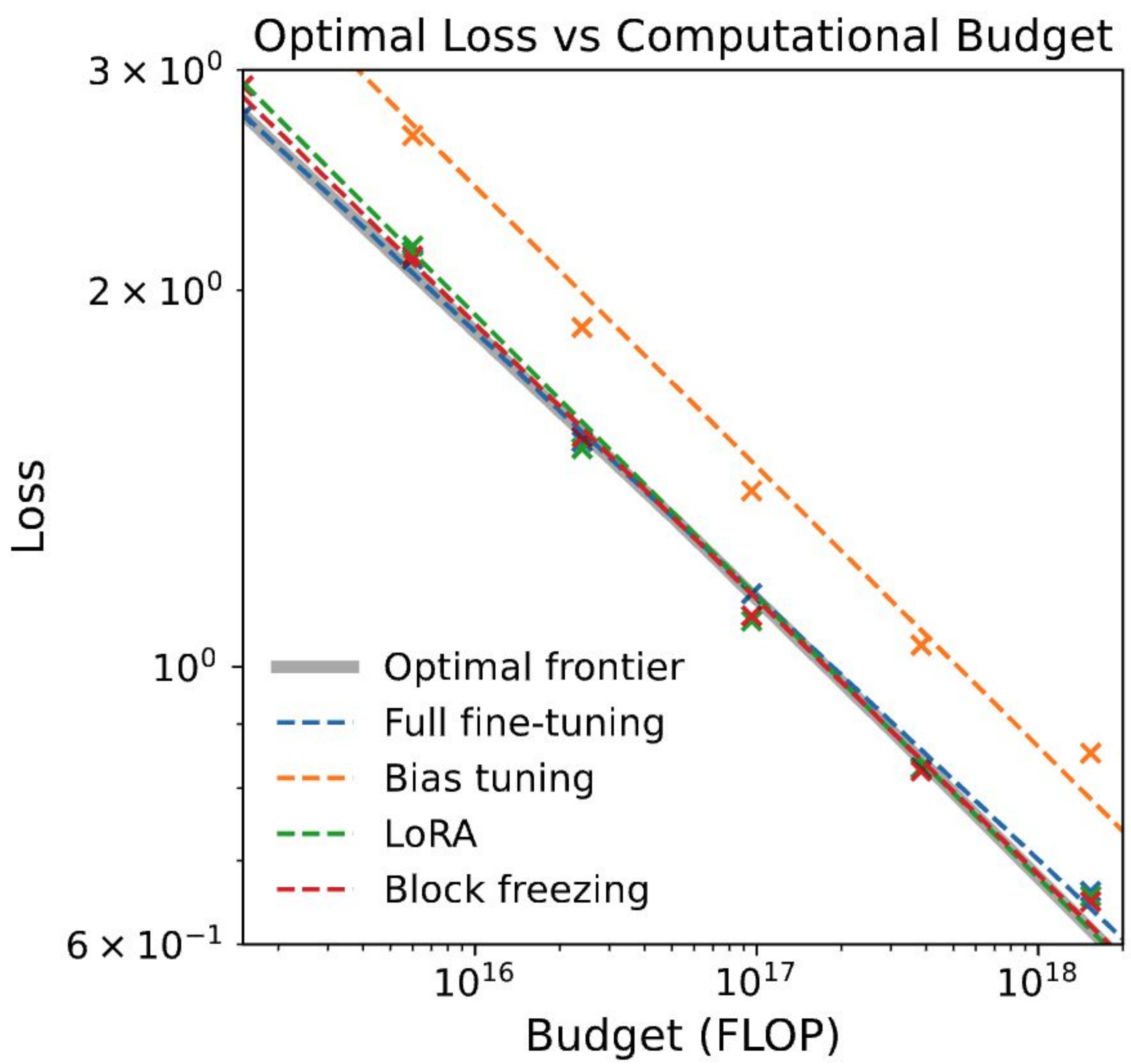
Updatable parameters

$$C = 2N_F D + 2N_B D + 2N_U D.$$

Forward parameters

Backward parameters

The compute-optimal recipe



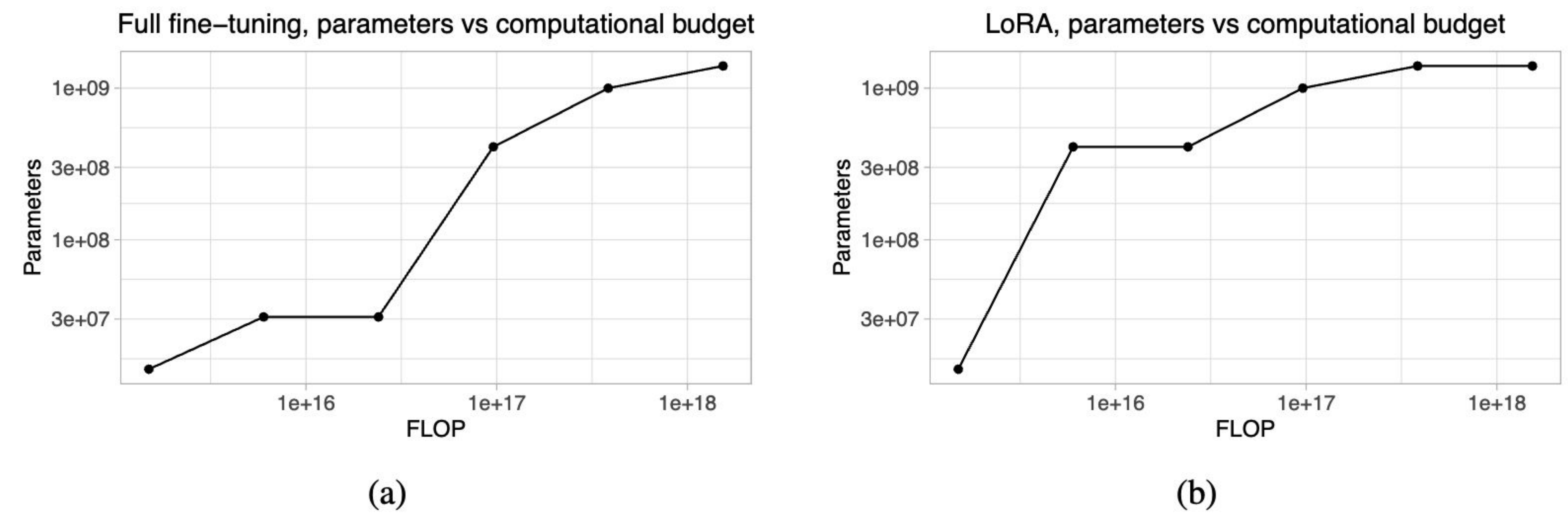


Figure 6: Optimal model size vs. computational budget for **(a)** full fine-tuning, and **(b)** LoRA.

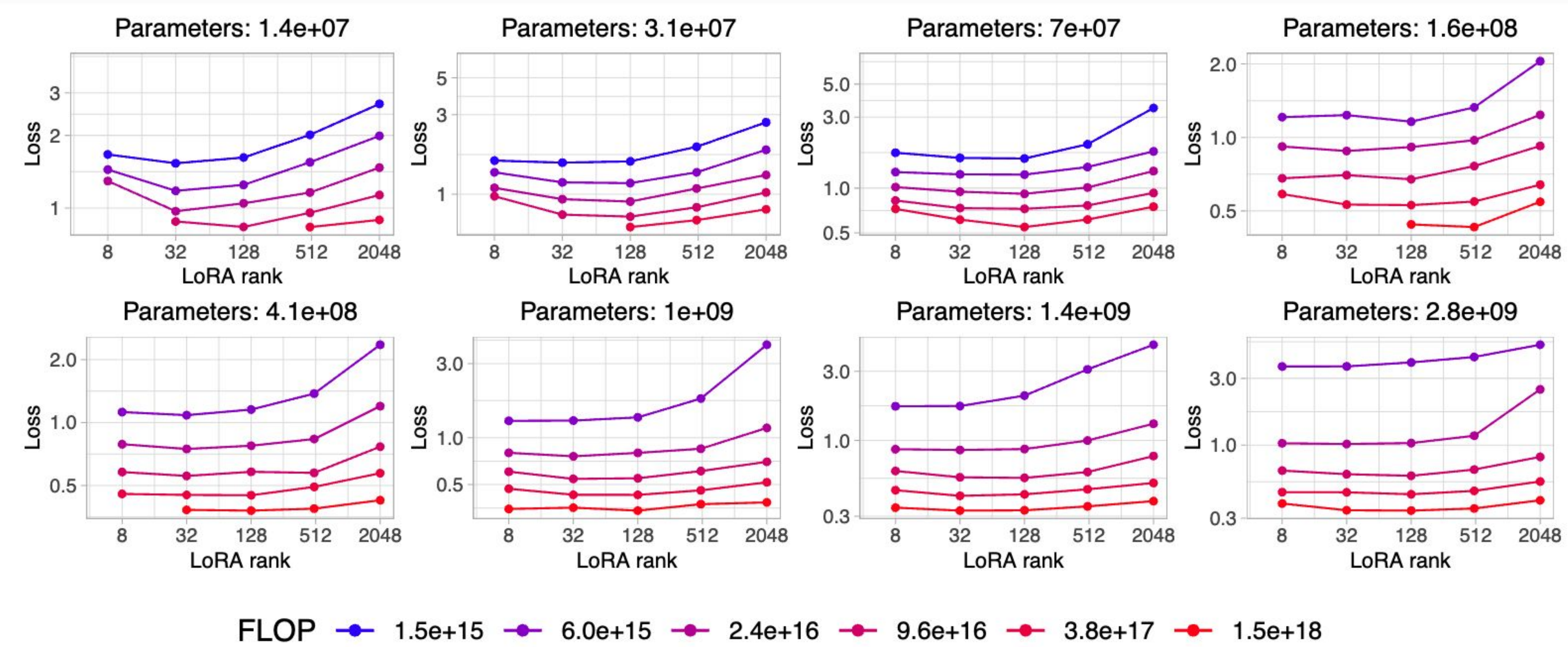


Figure 5: The effect of different LoRA ranks across all model sizes. Different colours signify different computational budgets. The inflected curves indicate that it is less beneficial to use a rank from either extremes of the spectrum (8 or 2048). The detrimental effect of the high rank of 2048 is stronger for lower computational budgets. Ranks of 32 and 128 result in the lowest loss overall.

Conclusion

- We derived scaling laws for training embedding models from decoder-only transformers. We found full fine-tuning and LoRA to be the most efficient methods at low and high computational budgets.

- Our scaling laws allow us to find the compute-optimal recipe for training embedding models, which reveals the optimal model size, data quantity, PEFT method and hyperparameters at a wide range of computational budgets.

Poster: Thursday 4:30PM - 7:30PM