# Understanding Compute-Parameter Trade-offs in Sparse Mixture-of-Expert Language Models


Harshay Shah[2*], Vimal Thilak[1*], Dan Busbridge[1], Alaaeldin El-Nouby[1], Josh Susskind[1], Samira Abnar[1*]

[1]Apple, [2]Massachusetts Institute of Technology (Work done while interning at Apple)

* Core contributors.


## FLOPs Vs Parameters

Under infinite data setting, scaling model capacity along with the training compute budget leads to performance improvements.

Current scaling law studies use parameter count as a proxy for model capacity. But this is not the only way to increase capacity.

Compute (FLOPs) "per example" is another way to increase model capacity (sparse MoEs, Chain-of-though, universal Transformers).

*It is crucial to jointly consider both **parameters** and **FLOPs per example** when deriving scaling laws.*

## Objectives

$$(N^*) = \arg\min_{N,S} \mathscr{L}(N; C)$$

$$\downarrow$$

$$(N^*, C_e^*) = \arg\min_{N, C_e} \mathscr{L}(N, C_e; C)$$

***Can we draw scaling laws for the optimal trade-off between parameter count and FLOPs per example?***

- To answer the question we study **Mixture-of-Experts Language Models**.
- **Sparsity**: the ratio of inactive experts to the total number of experts, which indirectly controls FLOPs per example in MoEs.

## Mixture-of-Experts

In MoEs, the compute per example $C_e \propto N_a$ and the number of active parameters $N_a \propto (1-S) \times N$.

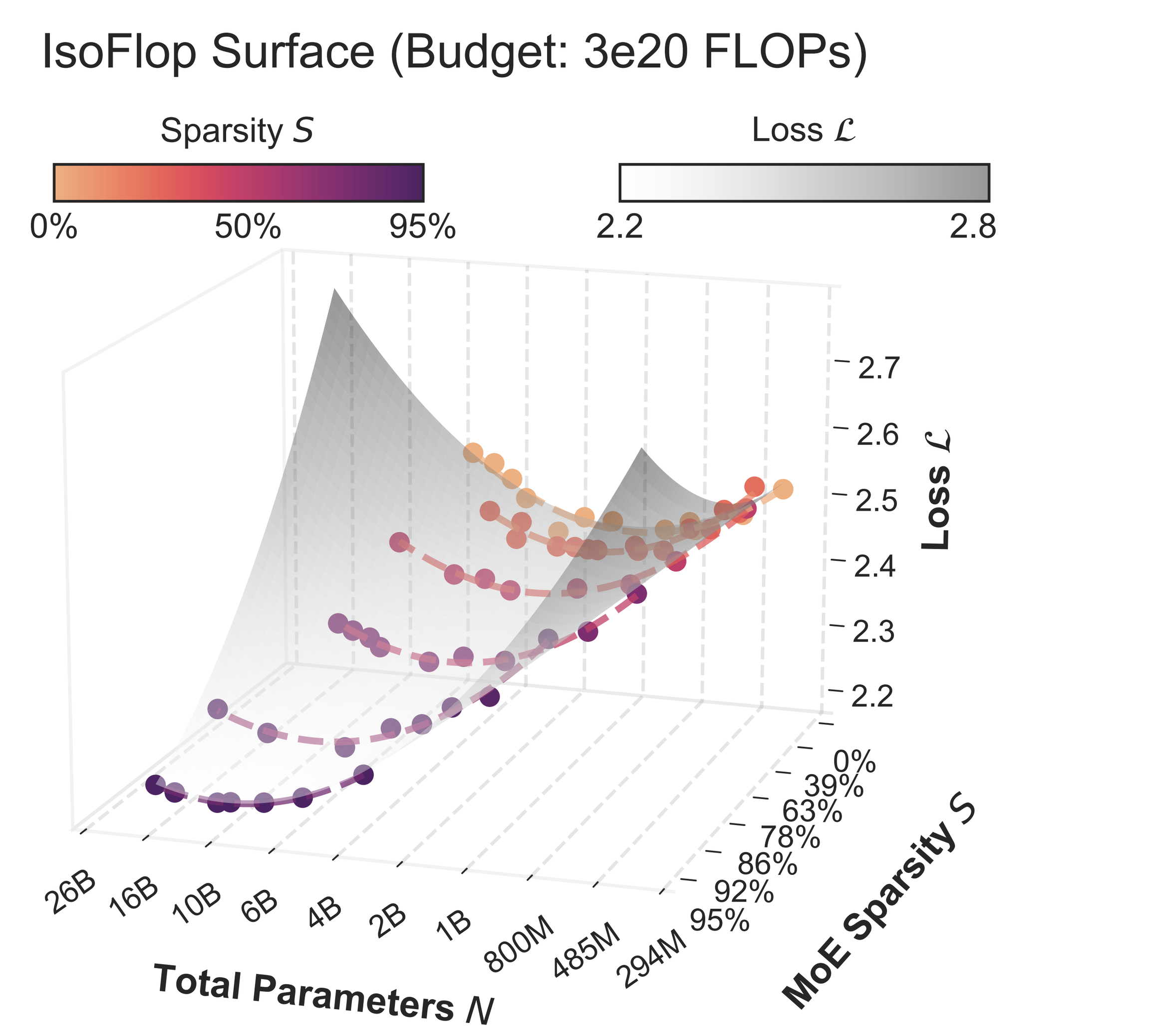
So $C_e \propto (1-S)$ where S denotes sparsity.

We study scaling laws of compute optimal models, jointly optimizing Sparsity and total parameters in MoEs:

$$(N^*, S^*) = \arg\min_{N,S} \mathscr{L}(N, S; C)$$

## Scaling Laws for Training Compute Optimal MoEs

### The Interplay between Parameter Count and Sparsity in MoEs

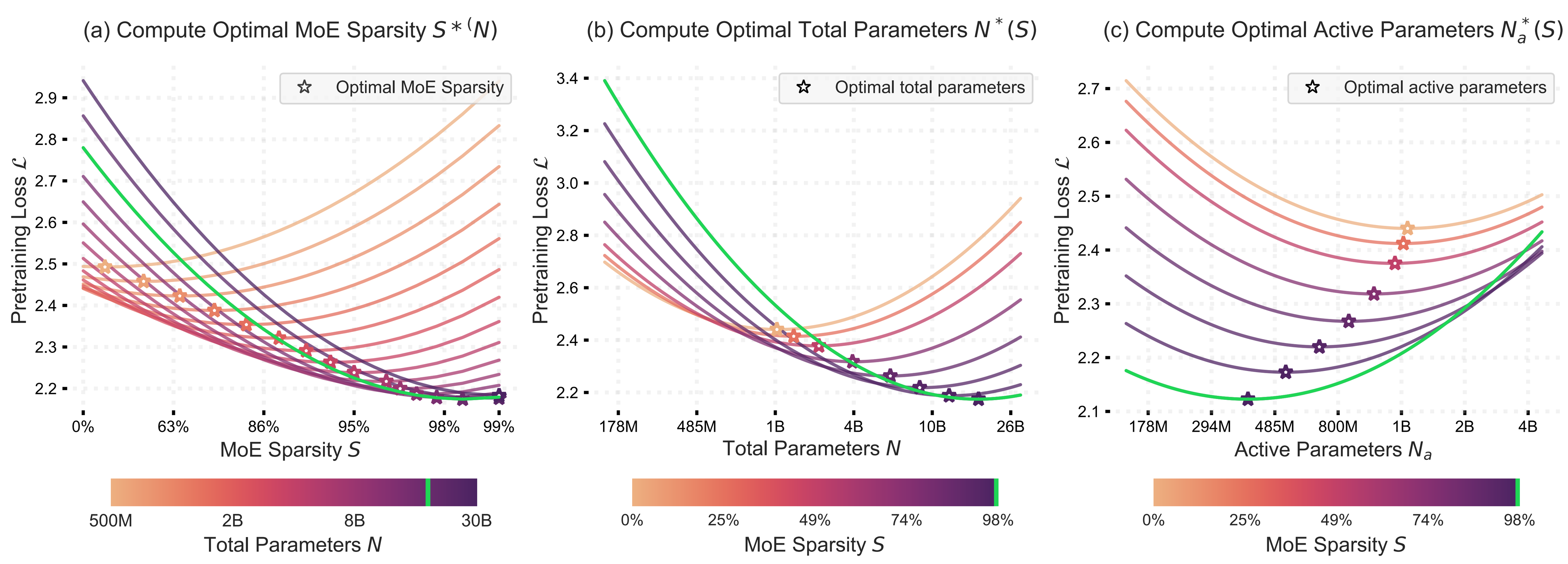



- Trained $\gtrsim 500$ Mixture-of-Experts Language Models:

  $S \in [0.0, 0.98] \quad C \in [3e+19, 1e+21] \quad N \in [60M, 15B]$
- Fitted 3d IsoFLOP polynomial surfaces to the data for each compute budget.

*Under fixed total training compute budget increasing sparsity in MoEs leads to smaller FLOPs per example, higher number of parameters, and lower pretraining loss simultaneously.*

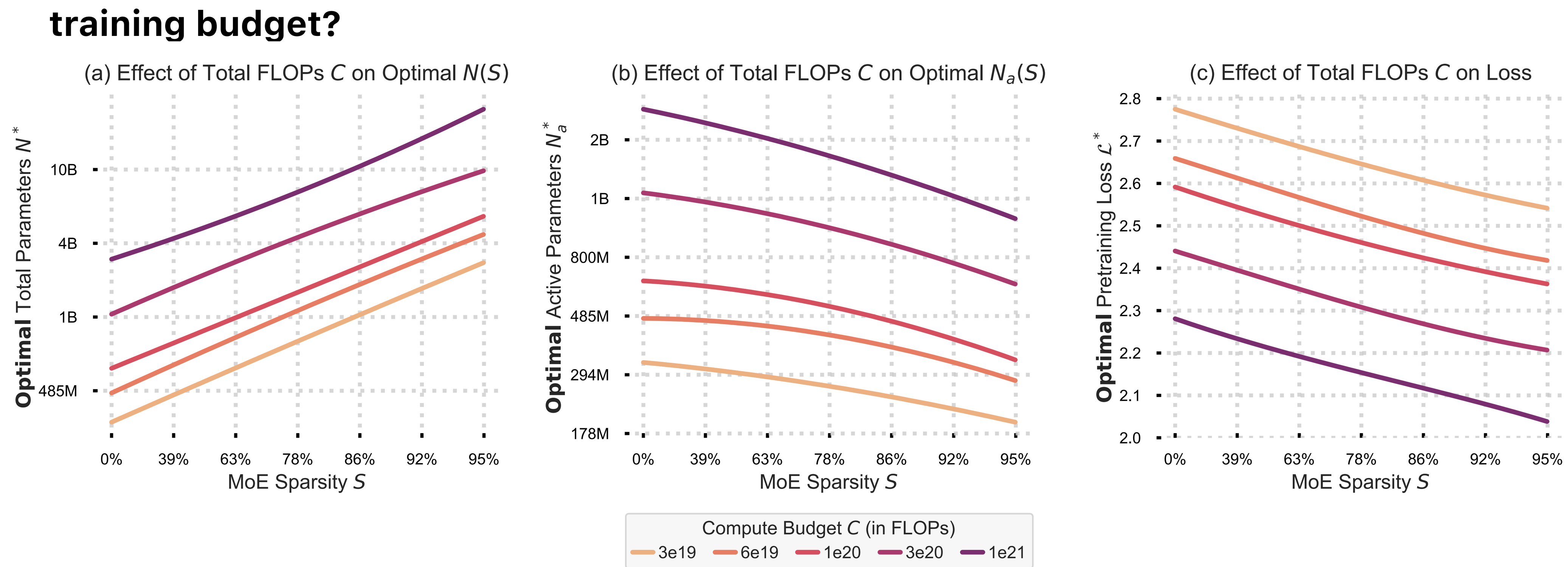
*Under conditions where memory, i.e., number of total parameters, is a constraint, we find that there is an optimal sparsity value that depends both on the total number of parameters and total training compute budget.*




$$S^* = \arg\min_{S} \mathscr{L}(S; N, C) \qquad N^* = \arg\min_{N} \mathscr{L}(N; S, C)$$

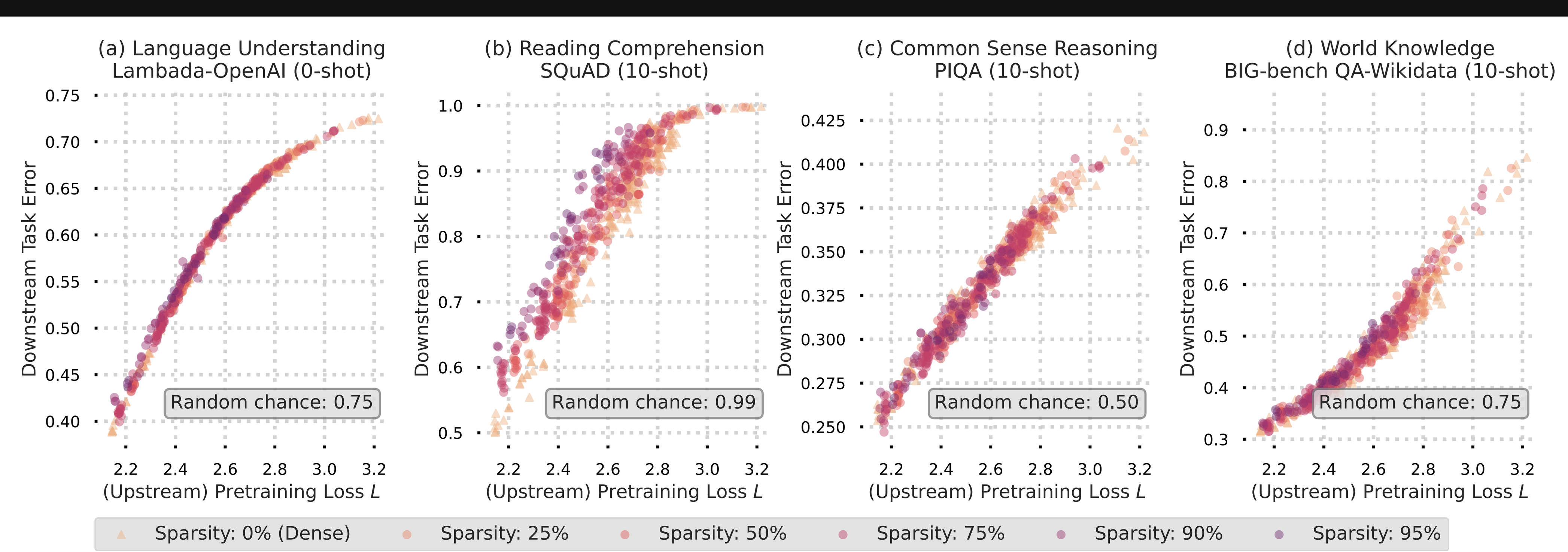
__IsoFLOP slices along Sparsity and Model Size__. We use fitted isoFLOP surfaces to analyze how sparsity S and model size N impact the loss L for a **fixed compute budget**. Observe that (a) the optimal sparsity S increases with increasing model size N and converges to 1 while (b) and (c) show that the optimal model size N and active parameter count Na increase and decrease respectively with increasing sparsity levels.

### Does the recipe for optimally increasing model capacity change as we scale up the training budget?




*We observe no diminishing effect of sparsity as we increase total training FLOPs.*

## Impact of Sparsity on Transfer




*Denser models perform better on certain types of task that may rely on deeper processing of the input vs the knowledge stored in the parameters of the model. This indicates the important role of FLOPs per example in increasing the capacity of the model during inference.*

## Conclusions

**Total parameter count** has a more significant role during at pretraining: when total training FLOPs is fixed, optimal strategy is to train larger sparser model with fewer FLOPs per example.

**FLOPs per example** seems to be more important during **inference** for specific types of tasks.

MoEs are efficient both in pertaining via improved capacity as well as inference via smaller number of active parameters. A potential benefit with lower cost is that MoEs may benefit from adaptive mechanisms to increase compute per example at inference, such as Chain of Though (CoT) reasoning.


*Questions or comments? Contact us at abnar@apple.com | vtluck@apple.com*