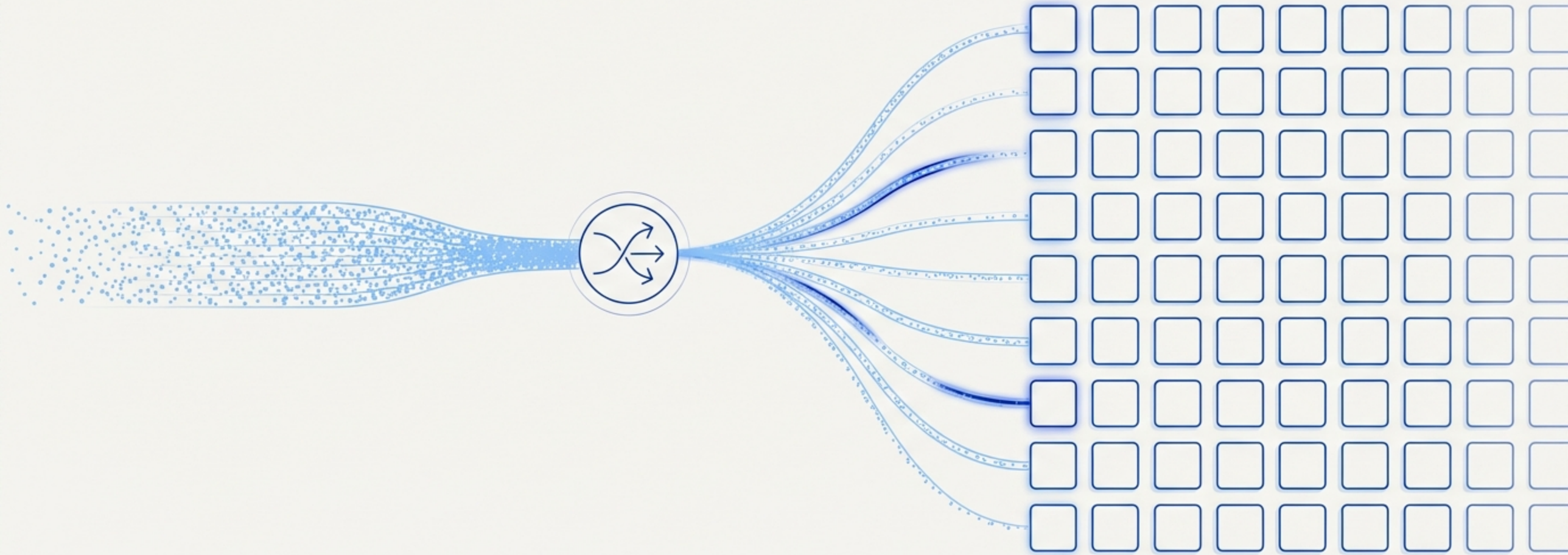
Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity

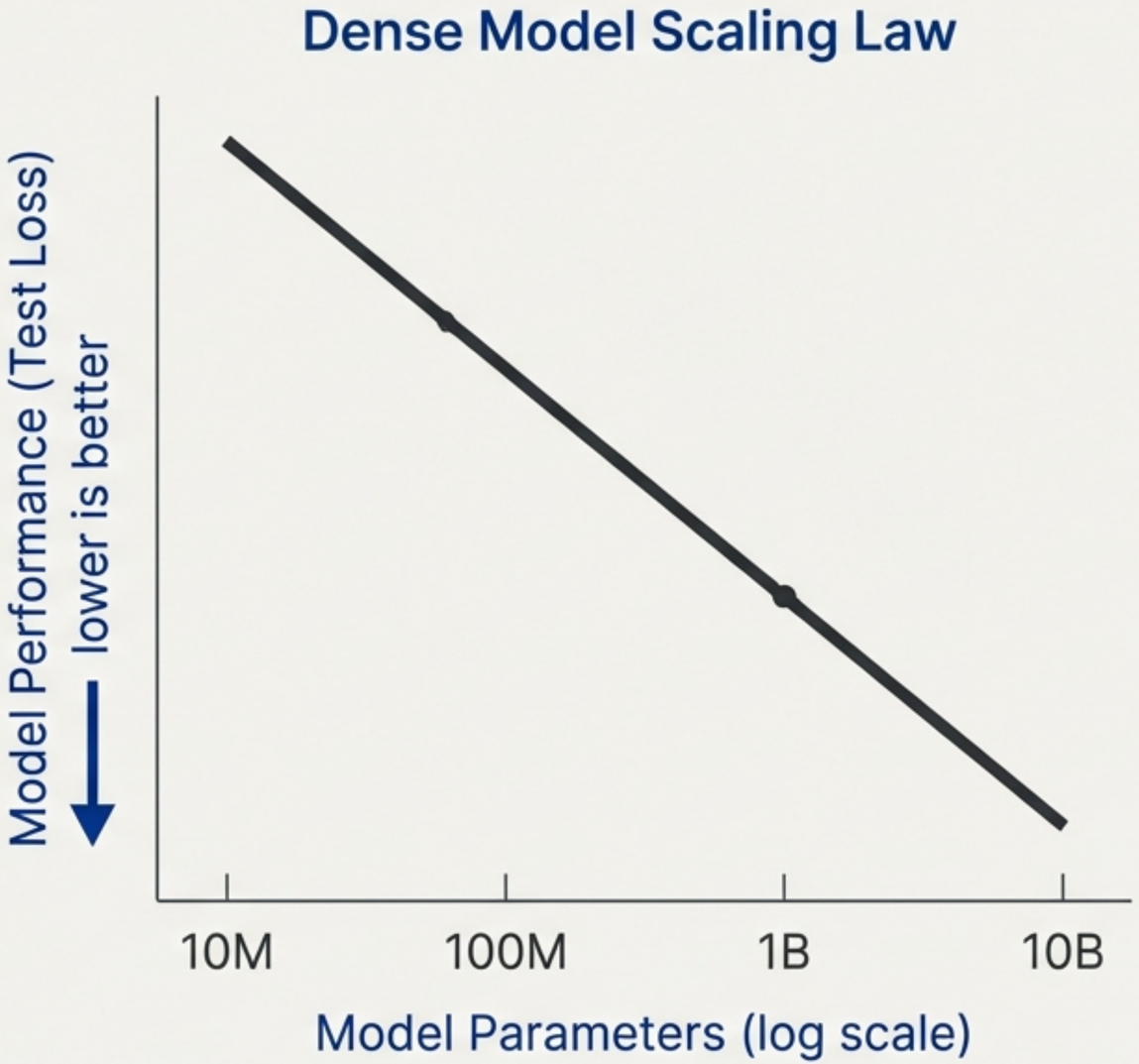
William Fedus, Barret Zoph, Noam Shazeer | Google Research | JMLR 2022



The Proven Path to Performance: Scaling Dense Models

The AI community has learned a “bitter lesson”: simple architectures backed by massive scale consistently outperform more complex algorithms.

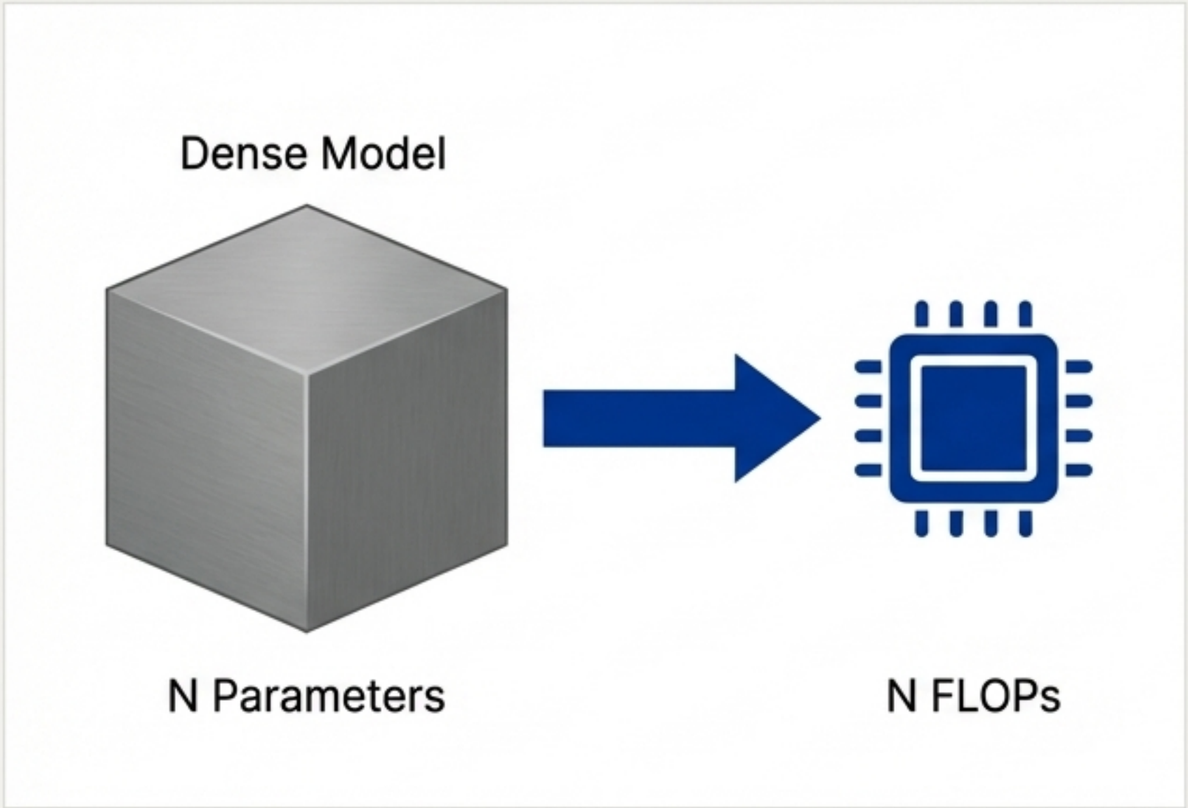
- This has led to a successful but computationally intensive “arms race” in model size (e.g., T5, GPT-3).
- For these dense models, every parameter is used for every input token.
- More parameters are directly coupled with more computation (FLOPs), creating a brute-force approach to better performance.

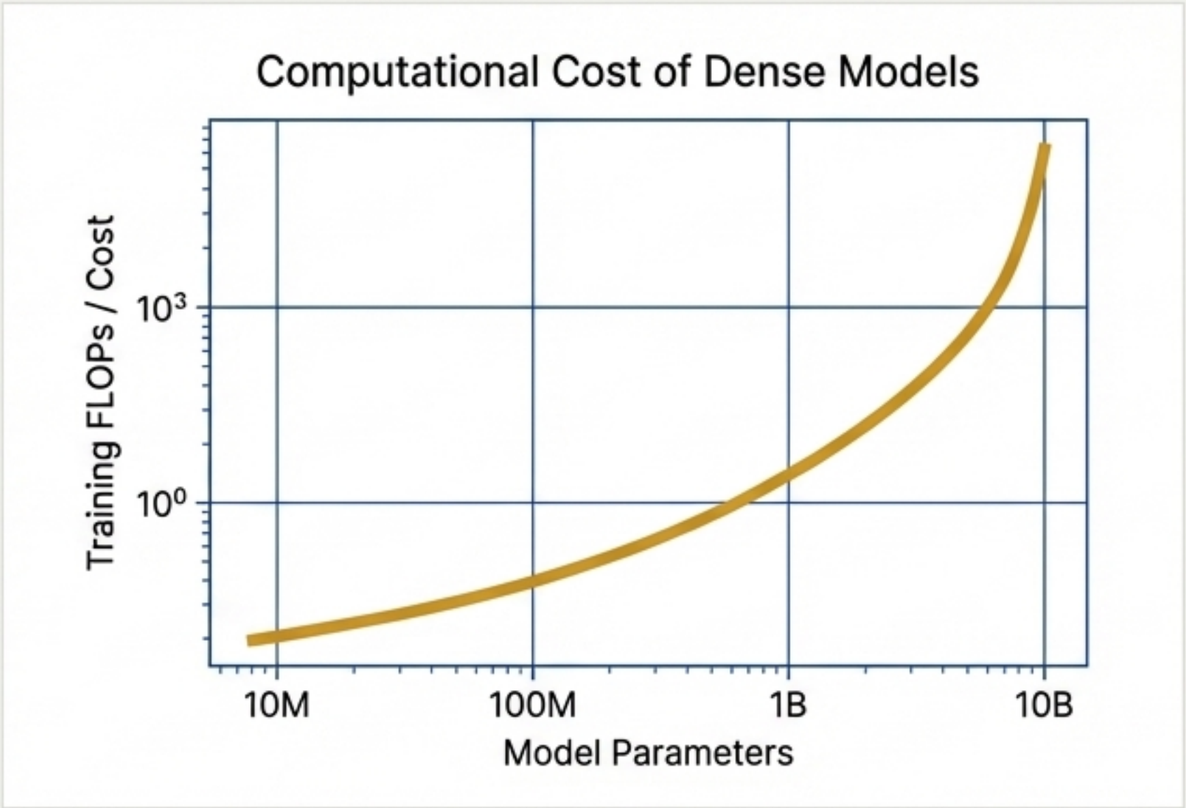


Dense Scaling Is Hitting a Wall of Unsustainable Cost

While effective, the brute-force approach of dense models is becoming computationally prohibitive and inefficient.

Can we decouple a model's parameter count from its computational cost per example?





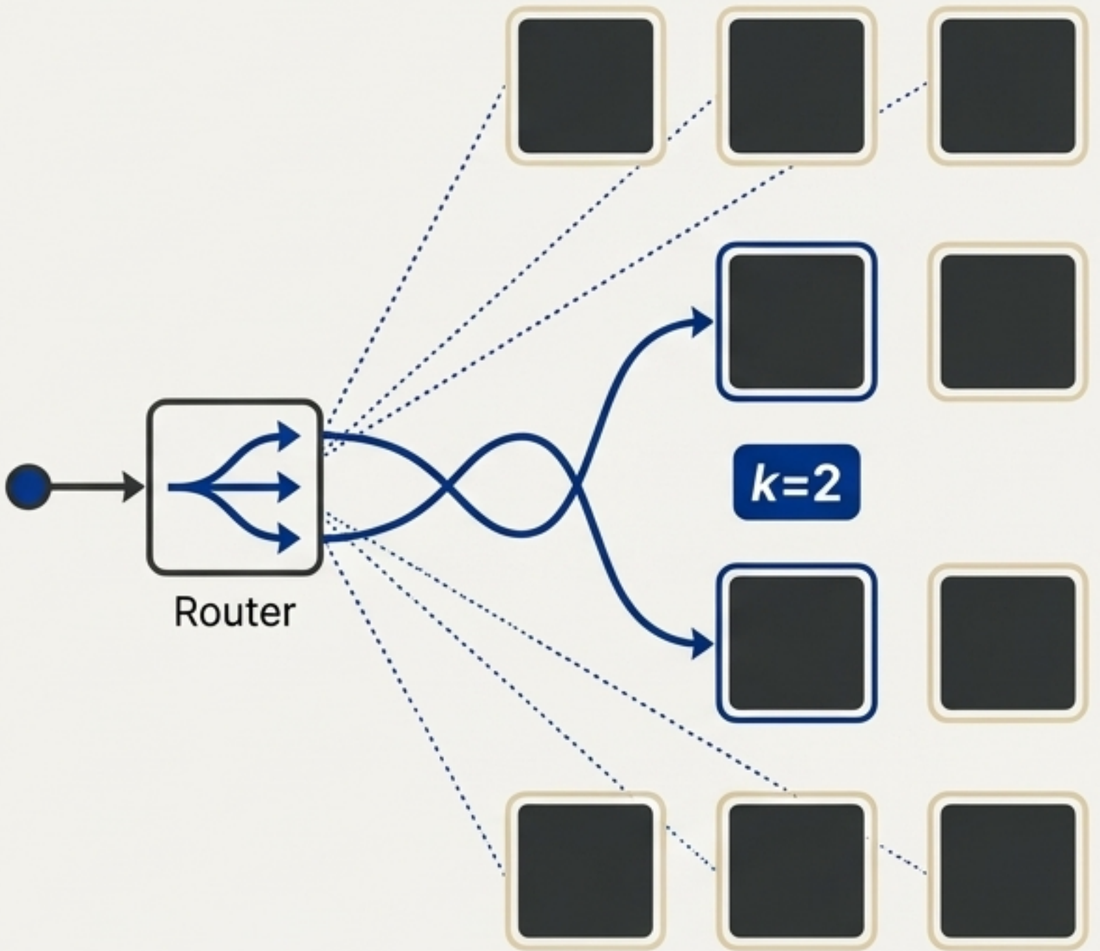
An Old Idea Promised a Way Forward: Mixture-of-Experts (MoE)

The Concept: Instead of one giant network, use a collection of smaller “expert” sub-networks. For each input, intelligently route it to only the most relevant experts.

The Promise: Achieve a massive parameter count (more knowledge capacity) with a constant computational cost (sparsely-activated model).

The Reality: Widespread adoption was hindered by significant flaws:

- **Complexity**: Routing to multiple experts (`top-k`) was algorithmically complex.
- **Instability**: Prone to training issues, often requiring full `float32` precision.
- **Communication Overhead**: Shuffling data between multiple experts was inefficient.



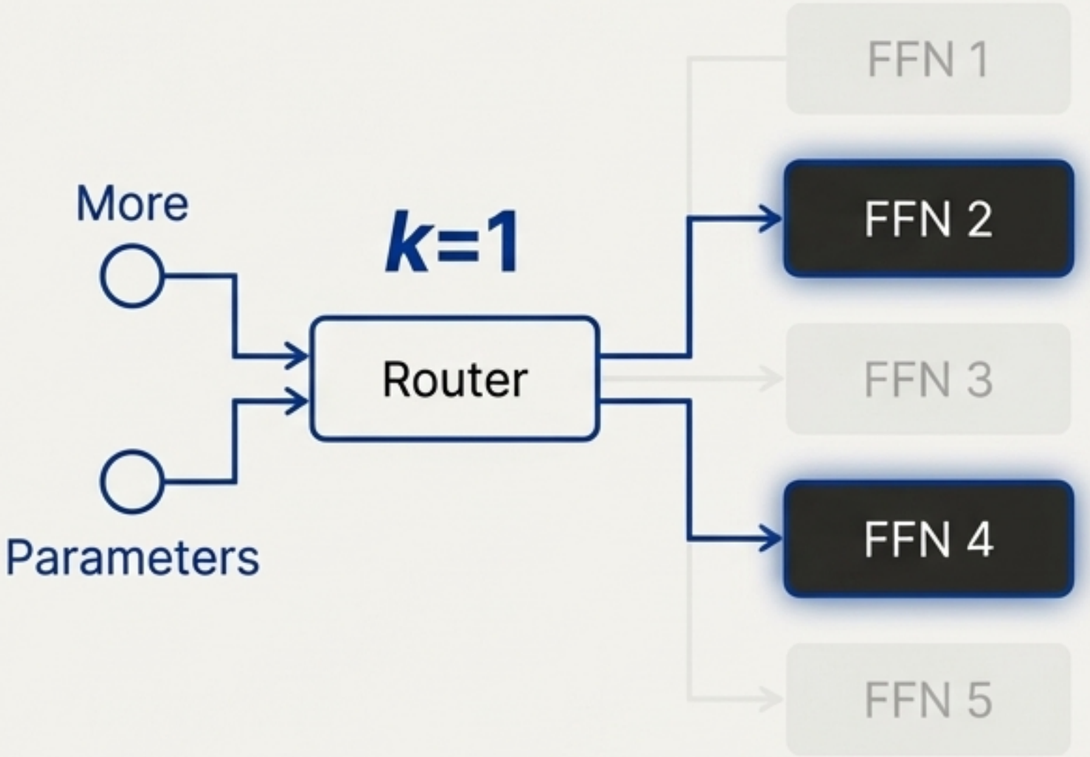
The Switch Transformer: Radical Simplicity Unlocks Sparsity

The Key Innovation: Replace the dense Feed-Forward Network (FFN) layer with a sparse Switch FFN layer.

The Breakthrough Idea: Route each token to only a *single* expert ($k=1$), contrary to prior work that assumed `$k > 1$' was necessary for learning.

Immediate benefits of k=1 routing:

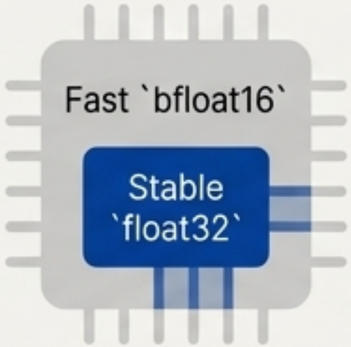
- Router computation is reduced.
- Communication overhead is minimized (token data sent to only one destination).
- Expert batch capacity is used more efficiently.



Three Key Techniques Make the `k=1` Approach Robust

1. Load Balancing Loss

An auxiliary loss is added to encourage the router to distribute tokens evenly across all experts. (Uses an $\alpha = 10^{-2}$ coefficient).



2. Selective Precision Training

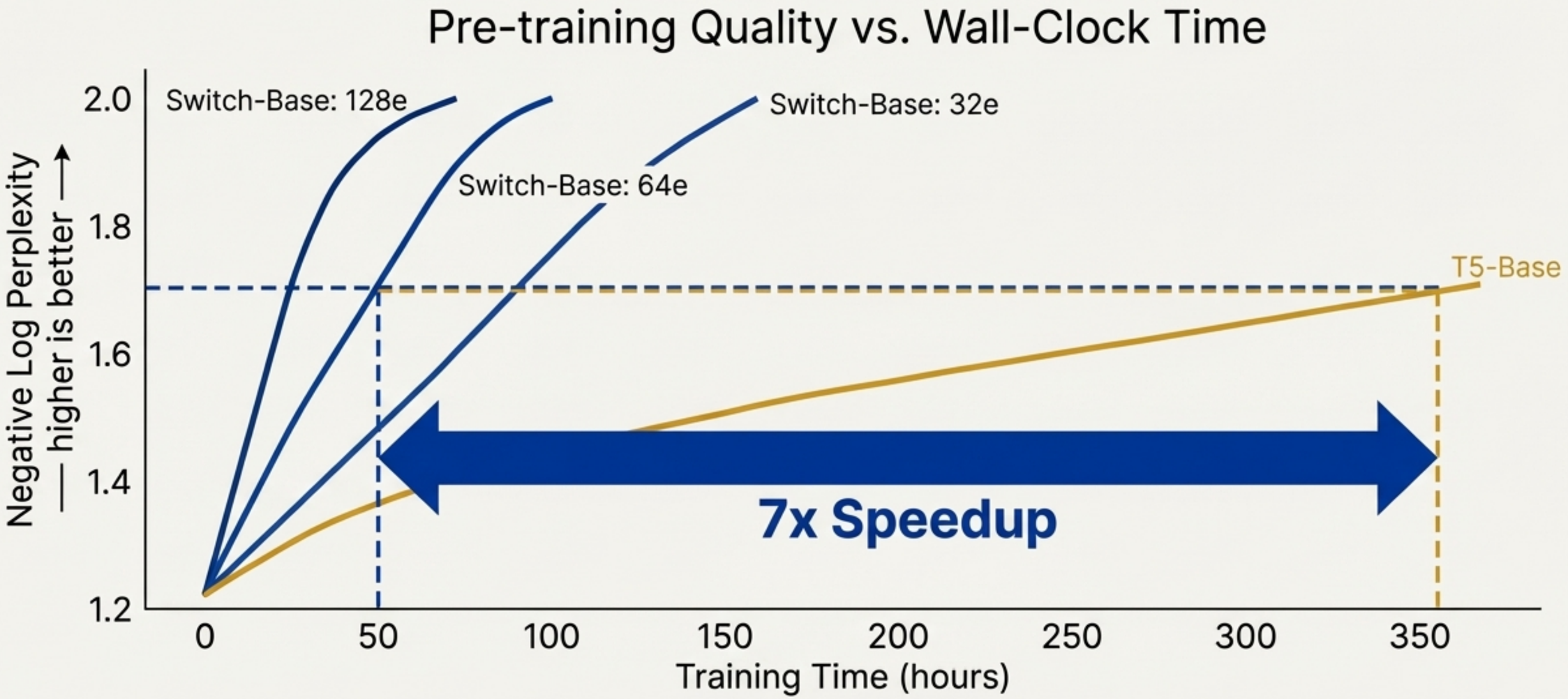
Solves instability by casting only the local router computations to `float32, while all other operations and expensive communications remain in fast `bfloat16'. This provides stability without sacrificing speed.

3. Improved Initialization & Regularization

Weight initialization scale is reduced by 10x to improve stability. A high 'expert dropout' rate is used during fine-tuning to prevent overfitting from the massive parameter count.

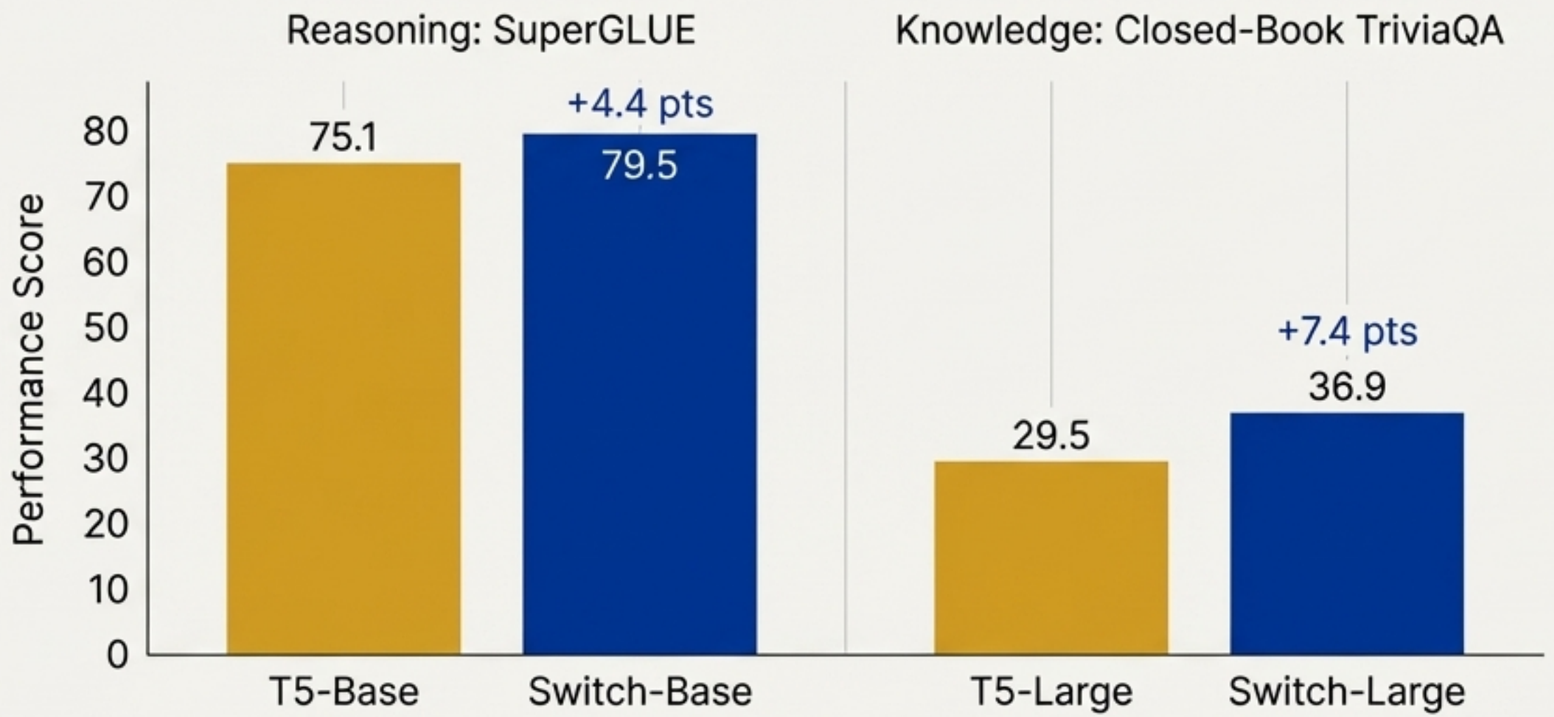
The Result: Up to 7x Faster Pre-Training for the Same Compute Budget

Core Finding: For a fixed amount of time and computational hardware (32 TPU cores), Switch Transformer models learn dramatically faster than the dense T5-Base baseline.
Key Stat: The 64-expert Switch-Base model achieves the same quality as T5-Base in one-seventh of the time.



Pre-Training Gains Translate to Strong Downstream Performance

Core Finding: When fine-tuned on NLP benchmarks, FLOP-matched Switch models consistently outperform their dense T5 counterparts.
Conclusion: This validates that the larger parameter count leads to more capable models, not just faster pre-training.

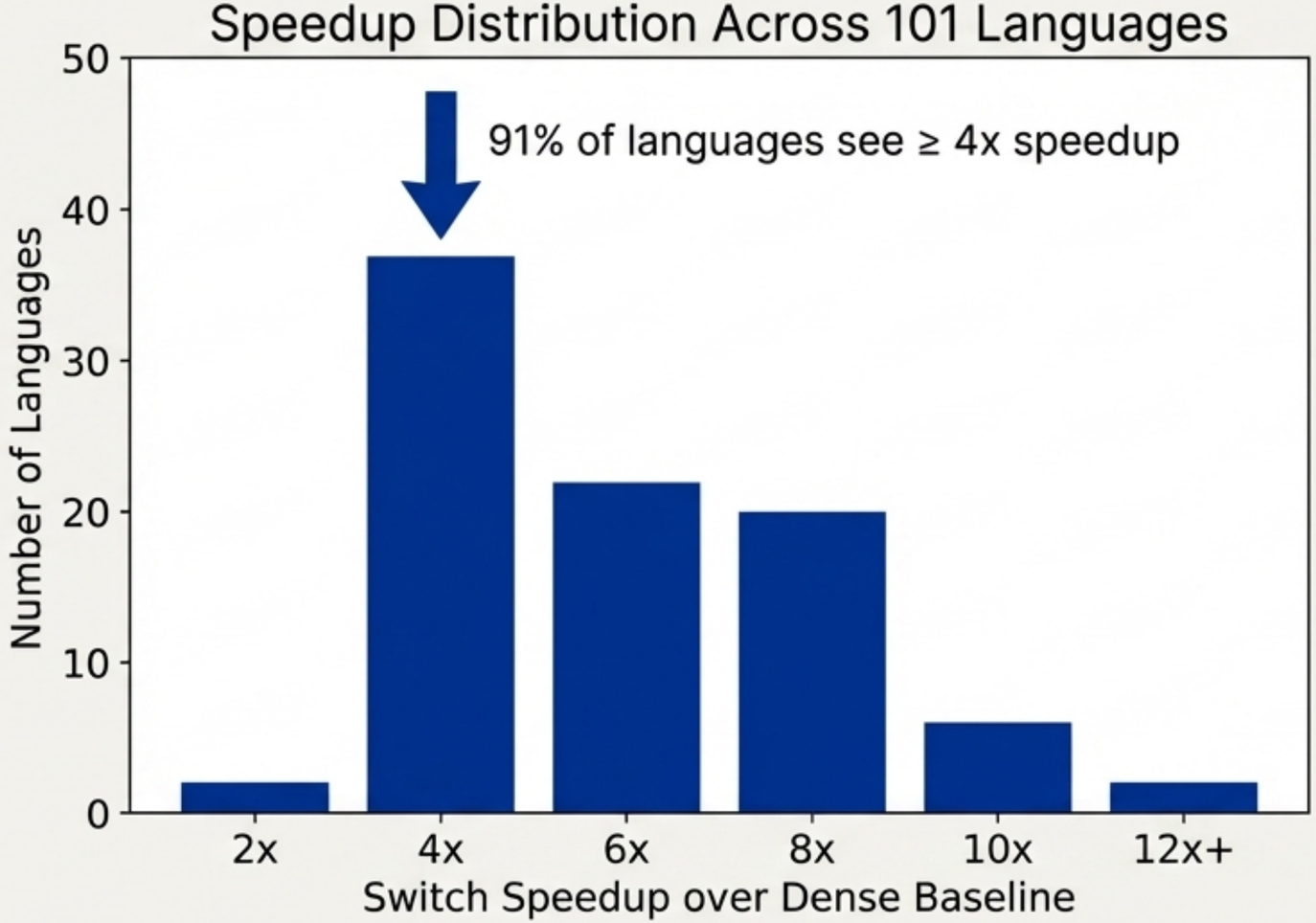


The Architecture Excels in a Massively Multilingual Setting

The Experiment: A Switch model (mSwitch-Base) was trained on the mC4 dataset, covering 101 languages.

The Results:

- Compared to the dense mT5-Base, the Switch model showed improved performance on **every single language**.
- For 91% of languages, the Switch model achieved at least a **4x speedup** in reaching the baseline's final quality.



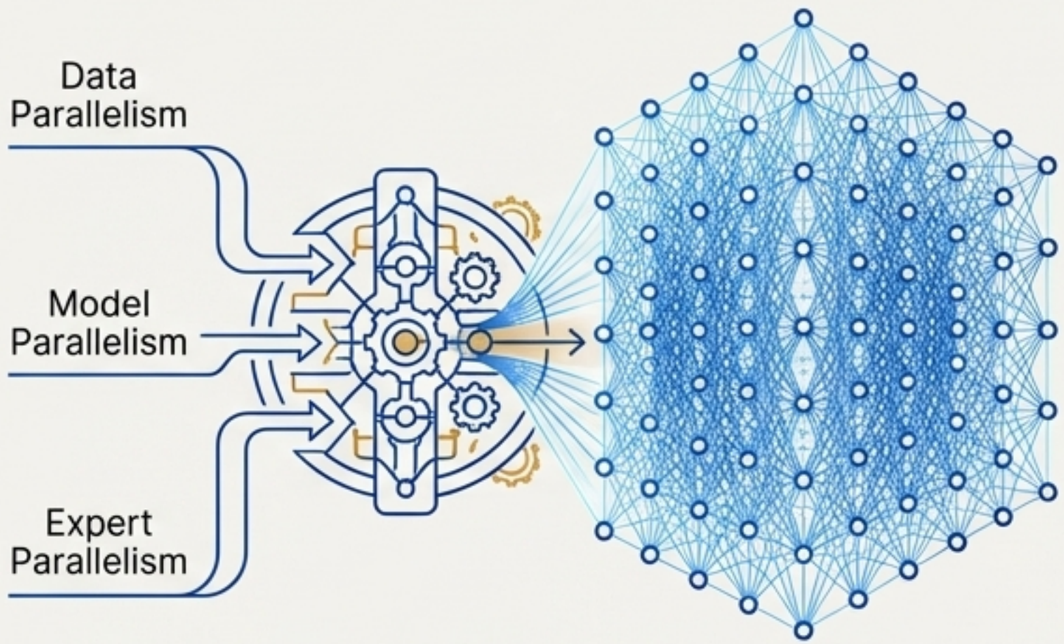
This Efficiency Unlocks Unprecedented Scale: The Trillion-Parameter Model

Core Achievement: By combining expert, model, and data parallelism, the authors designed and trained the largest language models to date.

The Models:

- **Switch-C**: A **1.6 Trillion** parameter model using 2048 experts.
- **Switch-XXL**: A 395 Billion parameter model, FLOP-matched to T5-XXL.

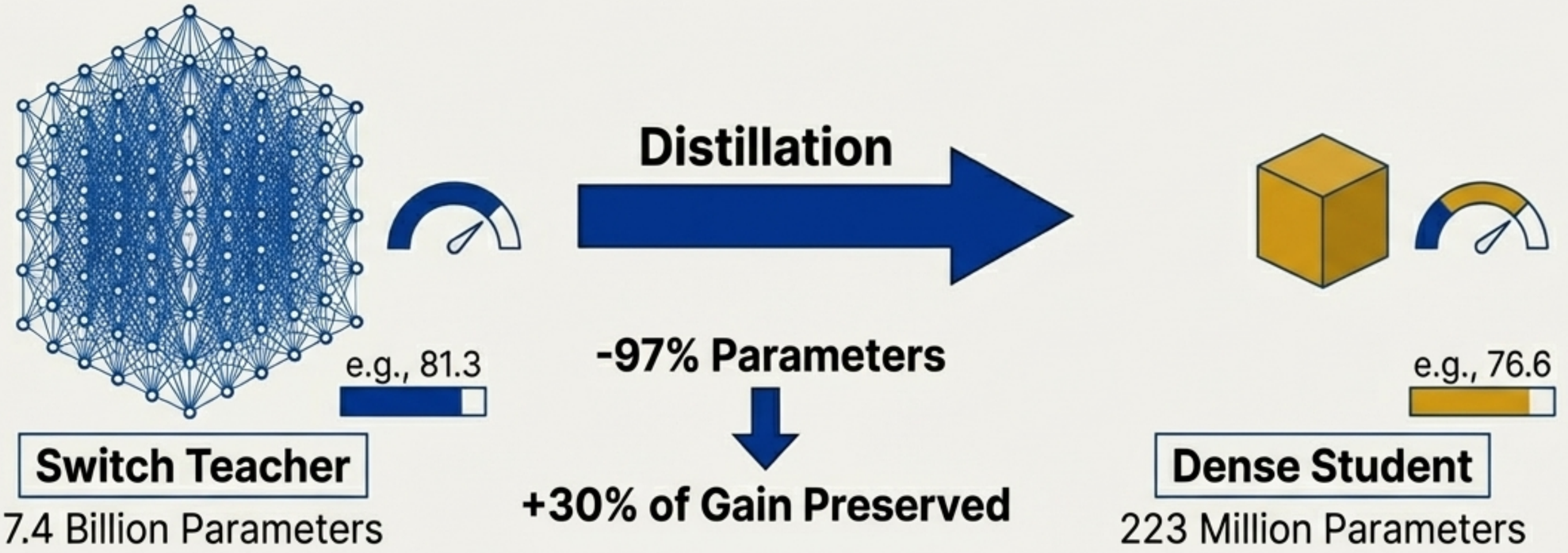
The Payoff: These models achieved a **4x speedup** over the 11-billion parameter T5-XXL, reaching its performance in a quarter of the time on the same hardware.



1.6 TRILLION PARAMETERS

Knowledge from Giant Sparse Models Can Be Distilled for Deployment

The Problem: A trillion-parameter model is impractical for most real-world applications.
The Solution: Use the large, sparse model as a 'teacher' to train a much smaller, dense 'student' model.
The Result: It is possible to compress model size by over **99%** while preserving nearly **30% of the quality gain** over a standard baseline. (e.g., 7.4B params to 223M).



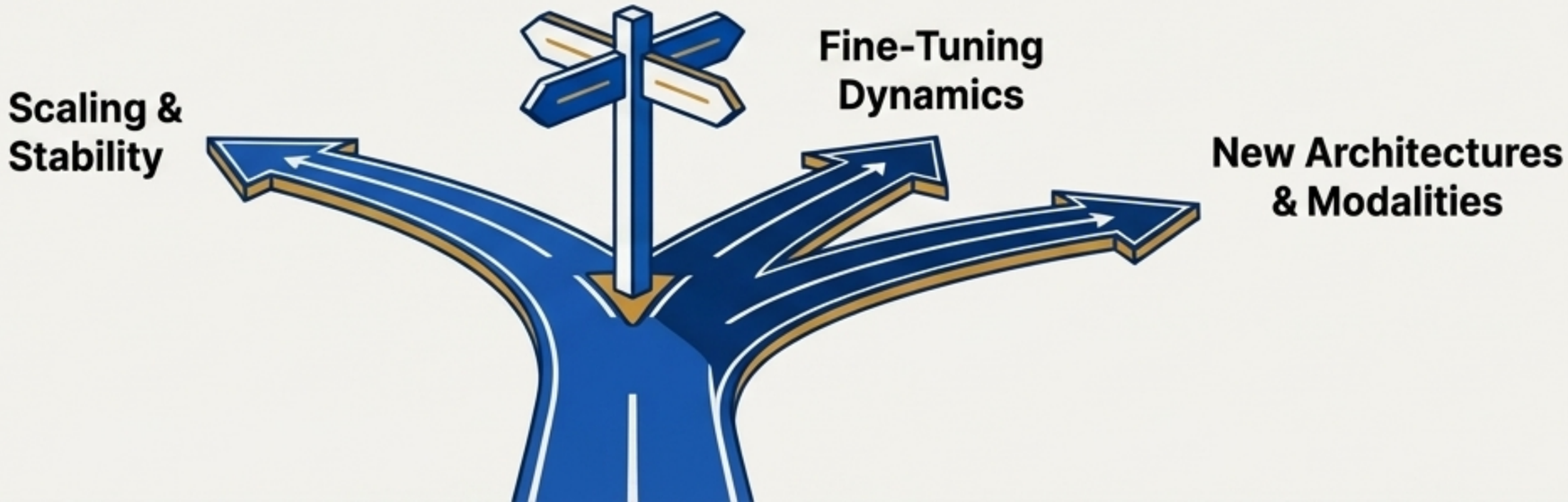
Open Challenges and Future Directions

Limitations

- **Stability at Extreme Scale:** The most FLOP-heavy model (Switch-XXL) still encountered sporadic instabilities.
- **Upstream vs. Downstream Performance:** The relationship between pre-training quality, parameter count, and FLOPs per token is complex. The 1.6T parameter Switch-C sometimes underperformed the smaller but more compute-heavy Switch-XXL on downstream tasks.

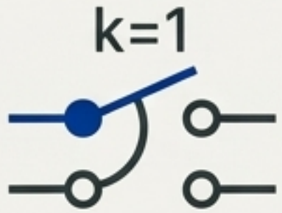
Future Work

- Applying Switch layers to other parts of the Transformer (e.g., attention).
- Exploring heterogeneous experts (different sizes/functions).
- Expanding to other modalities like vision and speech.



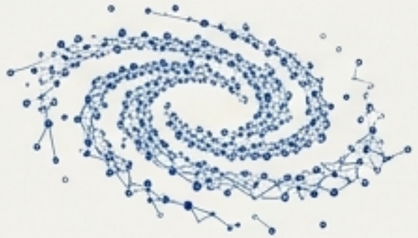
Switch Transformers Redefine the Speed-Accuracy Curve in AI

- **The Problem**: Scaling dense models is effective but computationally unsustainable.
- **The Solution**: A simplified, `k=1` Mixture-of-Experts architecture that is stable, efficient, and easy to implement.
- **The Impact**: Massive pre-training speedups (7x+), strong downstream performance, and a proven path to trillion-parameter models, fundamentally improving the economics of AI scaling.



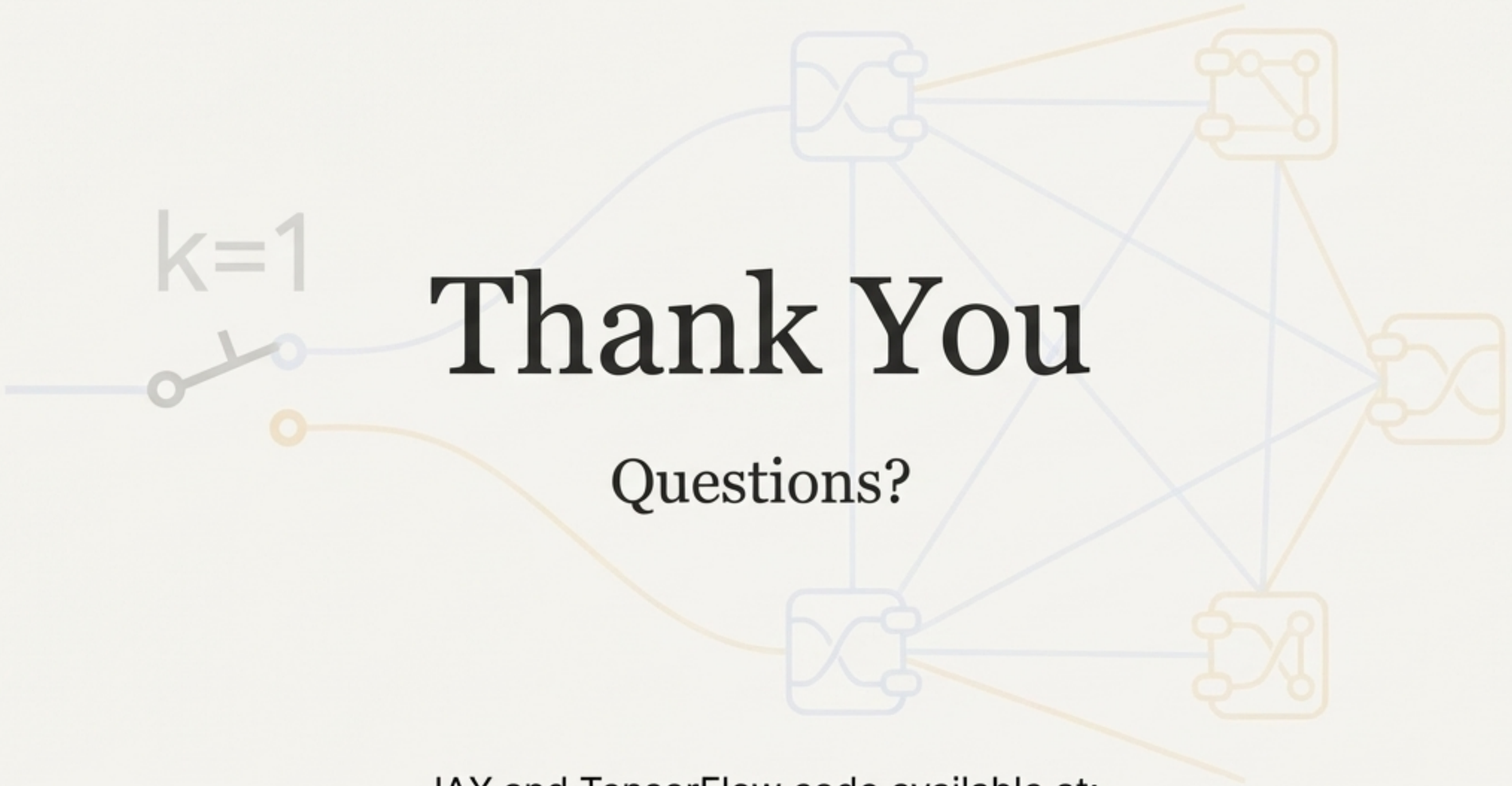
Radical Simplicity

Training Efficiency



Massive Scale

Downstream Performance



Thank You

Questions?

JAX and TensorFlow code available at:
github.com/google-research/t5x
github.com/tensorflow/mesh