TECHNICAL DOSSIER // DEEP LEARNING ARCHITECTURE & LLM INFERENCE

Mixture-of-Depths (MoD) Transformer Compute Allocation

Dynamic Layer Participation, Router Gating Topologies, and Inference Speedups

Published: October 30, 2026 | Distribution: Open Engineering Community | Document ID: LT-2026-B16-08

EXECUTIVE SUMMARY & MICROARCHITECTURAL SCOPE

Dense transformer models allocate an identical compute budget (FLOPs) to every token across all layers, squandering computational resources on predictable tokens like punctuation and common syntax. Google DeepMind's Mixture-of-Depths (MoD) dynamically routes tokens through transformer blocks using top-k capacity routers, allowing unselected tokens to bypass self-attention and MLP blocks via residual connections. This reduces inference FLOPs by up to 50% with zero loss in language modeling perplexity.

Model Architecture	Standard Dense Transformer	Mixture of Experts (MoE)	Early Exit Transformers	Mixture of Depths (MoD)
Compute Allocation	Fixed Uniform per Token	Sparse Layer Width (FFN)	Variable Depth per Token	Dynamic Layer Participation
Routing Target	None (All Tokens Active)	Expert FFN Blocks	Layer Exit Threshold	Self-Attention & MLP Blocks
FLOPs / Token (7B)	14.2 TFLOPs	7.1 TFLOPs (Active)	Variable (Unstable)	7.2 TFLOPs (Fixed Capacity)
KV Cache Footprint	100% Allocation	100% Allocation	Dynamic / Fragmented	Reduced (Skipped Layers)
Hardware Efficiency	High (Static GEMM)	Medium (All-to-All Comm)	Poor (Dynamic Branching)	High (Static Tensor Sizing)

1. The Compute Inefficiency of Uniform Transformers

In traditional autoregressive language models, computing a period, comma, or simple function word requires the exact same number of matrix multiplications as resolving a complex logical dependency or factual reasoning step. This compute uniformity forces foundational models to either under-compute difficult tokens or vastly over-compute trivial tokens, artificially inflating training costs and inference latency.

2. Top-k Router Mechanics & Capacity Factors

Mixture-of-Depths introduces a lightweight learned router before each transformer block. For a sequence of tokens X, the router emits scalar routing weights r = Sigmoid(W_r * X). Only the top-k tokens with the highest routing scores are admitted to the self-attention and feed-forward networks, while remaining tokens pass directly through the residual skip connection. Setting a fixed capacity factor C = 0.5 guarantees exact hardware memory allocation and predictable execution time.

3. Fused Triton Kernels & Autoregressive Decoding

During training, tokens can be gathered and scattered using vectorized matrix operations. During autoregressive decoding, where tokens arrive one by one, MoD routers determine whether a token needs attention projection. For bypassed layers, key and value projections are omitted, eliminating unnecessary memory bandwidth consumption and shrinking KV cache traffic.

4. Mixture-of-Depths and Experts (MoDE)

Combining Mixture-of-Depths with Mixture-of-Experts creates a dual-sparse architecture known as MoDE. MoDE routers dynamically modulate both model depth (which tokens compute at all) and model width (which specialized expert computes them), unlocking extreme parameter efficiency and delivering up to 3x higher inference throughput on modern GPU clusters.

STRATEGIC ENGINEERING TAKEAWAYS & DEPLOYMENT CHECKLIST

- **Dynamic Layer Participation:** Allocates compute where needed, skipping up to 50% of transformer layers on simple tokens.
- **Predictable Hardware Capacity:** Top-k capacity bounding ensures static tensor shapes and zero branch divergence on GPUs.
- **KV Cache Bandwidth Reduction:** Skipped layers eliminate KV cache reads and writes during generation.
- **MoDE Compound Sparsity:** Synergizes with MoE to simultaneously compress transformer depth and width.