MoE at the Edge: Offloading and Small Mixture of Experts Models for Constrained Hardware

Tawakkul Labs · September 17, 2026

Abstract. A technical survey of two parallel lines of work: expert offloading, which moves mixture of experts weights between memory tiers so large models run on constrained devices, and small mixture of experts models, which compress the architecture itself to sub billion scale. It covers the routing consistency caveat, predictive offloading, adaptive precision, expert merging, the on device MoE scaling law, and the honest open question of whether MoE actually helps on consumer hardware, all examined for what it means on African hardware budgets.

Introduction

The frontier has converged on a paradox. Mixture of experts is the default architecture for capable models, because it activates only a fraction of its parameters per token. But the total parameter count keeps growing, and total parameters are what a memory budget pays for. A 2.4 trillion parameter model with 95 billion active per token is efficient in compute and brutal in footprint. The question that matters for most of the world is not whether it is efficient, it is whether any of it can run on the hardware people actually have.

Two lines of work answer that question from opposite directions. Expert offloading keeps the big model and moves weights down the memory hierarchy, so a fraction of the experts lives in fast memory and the rest runs on CPU or loads on demand. Small mixture of experts models attack the other end, shrinking total parameters to the point where the whole model fits on a phone. This paper surveys both, the caveats that keep biting, and what each means on the four hardware tiers defined in our audit of local AI for rural and informal sectors.

What expert offloading is

A sparse MoE layer routes each token to a small subset of experts. The expert weights are the bulk of the model, and they are the part that must be present for a token to compute. Offloading systems exploit the gap between what must be resident and what can be deferred: a subset of experts is cached in fast memory, the rest lives on slower storage and is moved to CPU or loaded on demand when routing calls for it.

The taxonomy of the technique is stable across the literature. There is expert caching, where the most used experts stay resident. There is CPU offload, where idle experts sit in host memory and compute

there. There is load on demand, where experts are fetched from disk or across nodes when a token activates them. And there is the crucial research finding of the last two years: the benefit is not uniform.

The routing consistency caveat

The strongest result in the offload literature is a negative one. The ICLR 2026 paper "Not All Models Suit Expert Offloading" showed that whether offloading works depends on how consistent a model's routing is. If a token routes to the same experts across similar inputs, then predicting the next experts and prefetching them works, and offload is nearly lossless. If routing is chaotic, the cache misses constantly, latency balloons, and the offload buys nothing. Local routing consistency is therefore the property that determines whether an offload system helps at all, and it is not something a system can fix. It is a property of the model.

This reframes the engineering problem. The offloading system is not the whole story. The model's routing behavior decides the ceiling. Any practical deployment must measure routing consistency before choosing an offload strategy, and models that route inconsistently are better served by the other line of work entirely.

Offload techniques in practice

Given a model with consistent routing, the techniques stack. ExpertFlow uses predictive offloading, learning which experts a token will need and prefetching them before they are required, which turns the memory bottleneck into a scheduling problem. MoE-APEX adds adaptive precision, keeping experts at full precision in fast memory and quantizing the offloaded majority, trading a controlled quality loss for a large memory win on edge devices. The adaptive expert merging work out of TU Berlin goes further, merging experts at inference time instead of loading them, which removes the offload cost for models whose experts are redundant. At the extreme, OD MoE eliminates the expert cache entirely, distributing expert loading across nodes and predicting activations layers ahead, reaching most of the cached performance with a third of the memory.

The common thread is that the community has stopped treating offload as a blunt swap and started treating it as a per model scheduling and precision problem. That is exactly the kind of engineering that matters on constrained hardware.

The small MoE wave

The other direction compresses the model until offload is unnecessary. Small mixture of experts models keep the sparse architecture at sub billion scale, and the past two years produced both the evidence and the counter evidence. EdgeMoE pioneered on device MoE inference by holding non expert weights in memory and expert weights on external storage, an offload inside a small model. MobileMoE went

further and derived an on device MoE scaling law, jointly optimizing the architecture under mobile memory and compute budgets, and asking whether MoE even pays off below a billion parameters.

That question is live. The 2026 paper "Does Mixture of Experts Actually Help Inference on Consumer and Edge Devices" tested the received wisdom directly, because MoE models advertise per token FLOPs comparable to a much smaller dense model, and that is not the same as faster on real hardware. Memory traffic, expert loading, and routing overhead can erase the theoretical advantage at small scale.

The open model wave gives the answer somewhere to live. OLMoE proved a fully open MoE can beat dense models of similar cost. And the extreme sparsity data from our State of AI Architecture 2026 survey shows the trend is accelerating: Qwen3.8 activates 4 percent of 2.4 trillion parameters, DeepSeek V4 Pro activates 3.1 percent, Tencent Hy4 activates 6.4 percent, and Nemotron Lightning activates 3 billion of 30 billion. Total parameters have stopped being a serving cost and become a quality reservoir, which is exactly why small MoE models are the interesting tier for local deployment.

Does MoE actually help

The honest answer is: it depends on the device, and the measurement is rarely what the marketing says. MoE reduces FLOPs per token, but FLOPs are not the constraint on a phone. The constraints are memory bandwidth, expert loading latency, and the cache behavior of routing. A small MoE model with 3 billion active parameters still touches a routing table and a fraction of the expert pool, and on a phone that can be slower than a dense model of the same footprint that keeps everything in fast memory.

The pragmatic verdict for constrained hardware is three way. Dense quantized models remain the safe default for small footprints, because there is no routing overhead and no offload. MoE models win when total parameters would otherwise blow the memory budget and routing is consistent. And offloading only pays when the model routes predictably and the storage is fast enough to hide prefetch. Each tier of hardware lands on a different point of that spectrum.

What it means on African hardware budgets

Mapping to the four tiers from our local AI audit changes the recommendation per tier. On a budget phone with 1 to 2 gigabytes of usable memory, the answer is dense quantized small models, full stop. MoE routing and expert loading are not worth the overhead at that scale. On a 2 to 5 gigabyte tier, small MoE models like the 3 billion active class become viable, and the extreme sparsity trend means a quality reservoir can ride in the same footprint. On a 5 to 16 gigabyte tier, expert offloading becomes genuinely useful, since a large MoE model can keep its active experts resident and offload the rest to storage, provided routing consistency is measured first. On the server tier, the full offload toolbox applies.

The strategic read for builders is that MoE changes the economics of quality but not the physics of memory. The devices most of the world uses get the small end of that trade, and the small end is where

the research is still genuinely contested. That is a useful thing to know before building.

What to watch next

Three threads will resolve the open questions. The first is routing consistency as a measurable model property that model card publishers start reporting, so offload decisions become data driven. The second is the on device MoE scaling law maturing into a practical recipe for the 1 to 4 gigabyte range, which is the mass market. The third is expert merging replacing offloading for models whose experts are redundant, which could make the whole memory hierarchy argument moot at the small end. Each of these is a concrete decision point for anyone building local AI for constrained environments.

Sources

- Liang et al. "Not All Models Suit Expert Offloading: On Local Routing Consistency". ICLR 2026.
- Fate: Fast Edge Inference of Mixture of Experts Models via Cross Expert Prediction. 2026.
- ExpertFlow: Efficient Mixture of Experts Inference via Predictive Offloading.
- MoE APEX: An Efficient MoE Inference System with Adaptive Precision Expert Offloading. 2026.
- Adaptive Expert Merging for Efficient MoE Inference at the Edge. TU Berlin, 2026.
- EdgeMoE: Empowering Sparse Large Language Models on Mobile Devices.
- MobileMoE: Scaling On Device Mixture of Experts. 2026.
- "Does Mixture of Experts Actually Help Inference on Consumer and Edge Devices". 2026.
- OLMoE: fully open mixture of experts language model.
- Tawakkul Labs. "State of AI Architecture 2026" and "Local AI Architecture for Rural and Informal Sectors".

Tawakkul Labs, Nairobi. Open source research. Published at https://tawakkul-labs.co.ke/research/007-moe-at-the-edge