MoE Offload on RTX 3090: The Curve Is Linear, Not a Cliff (2026)

August 3, 2026 · by Mark Bartlett

Download this post as PDF

InsiderLLM Weekly issue 16 – August 3, 2026

Every layer you push off the card costs roughly half a millisecond, and it keeps costing that all the way down — no knee, no cliff.

Quick Hits

- **A 30B MoE arrived from a German public consortium — no lab, no hyperscaler.** Soofi S 30B-A3B (arXiv 2607.09424) is a hybrid Mamba-Transformer MoE for German and English, out of Fraunhofer IAIS, DFKI, TU Darmstadt and two companies, coordinated by the KI Bundesverband on federal funding. They shipped intermediate checkpoints, the full data accounting and the eval code with it. **What's on Hugging Face is a gated preview, and the licence text isn't finished** — the permissive release is promised, not published. I don't have access and haven't put it on a card. If you do: what does it give you at full expert offload? Reply and I'll print it. 📖 The MoE primer.
- **Trinity-Mini's first two offload steps move nothing at all.** Headless on the 3090: 206.58 tok/s fully resident, 206.08 at `-ncmoe 1`, 205.91 at `-ncmoe 2` — three runs inside each other's error bars. Then `-ncmoe 3` drops it to 173.92. The GGUF header explains it: `afmoe.block_count` is 32, but only 30 of those blocks hold experts, and the two that don't are the two at the front — exactly the ones you offload first.
- **One thread per logical CPU cut our throughput in half.** With experts in system RAM the work is bound by memory bandwidth, not cores, so hyperthread siblings end up fighting over the same memory controller. `-t 12` on our 6-core 3090 box took it from 32.6 to **16.5 tok/s**, error bars three times wider. Nothing in the output flags it. The run just takes twice as long. 📖 Why your local LLM is slow.

There's a sentence on our Qwen guide I wrote in July and believed until Sunday. The table printed directly above it says it's wrong. Fixing it broke the rule underneath, and that took a third model.

The First Offloaded Layer Is the Cheapest One

Our Qwen 3.6-35B-A3B guide has carried an eleven-rung `-ncmoe` sweep since July: fully resident at 157.66 tok/s, every expert pushed to system RAM at 36.11, nine points in between. Below it sat a line I wrote and believed — moving one layer's experts off the card costs 11.2 tok/s, and that's the price of the offload path existing at all, before any question of how much data moves.

It's wrong, and the table directly above it says so.

Convert that column to milliseconds per token and the sweep is a straight line. Each offloaded layer adds about **0.53 ms**, and it keeps adding roughly that much for all forty of them. Fit a line through every rung and it accounts for **99.9%** of the variance. There's no knee, and no fixed cost for turning the feature on. The first layer costs 0.486 ms, **below** the 0.529 average. It's the cheapest layer we measured directly, not the most expensive.

The 11.2 tok/s is real. It's just what 0.53 ms looks like when you're sitting at 157 tok/s. Down at 36, that same 0.53 ms costs you 0.7 tok/s. Tokens per second is a reciprocal scale, so identical increments of work look enormous at the top of it and invisible at the bottom. That's the whole cliff.

The advice survives. The reason I gave for it doesn't. Two layers of "just a little headroom" costs about **1 ms per token, 14% of your throughput**, on a card that never needed to give any of it up. Offload because you have to, not because it feels safer.

📖 **The corrected sweep, with the ms/token column, is here.**

Wrong Twice, Fixed in Public

That sentence wasn't a one-off. It came from a rule this site asserted across more than a dozen pages: active parameters decide how fast a model runs, total parameters decide whether it loads at all.

The first half finally got tested. Two MoE models both labelled "3B active" should, by that rule, behave the same once their experts live in RAM. On my 3090, offloading every expert costs Trinity-Mini 28.98 ms per token and Qwen 21.40 — 35% apart, from labels predicting no difference whatsoever. That 35% is the marginal cost of offloading on one card, not the throughput you end up with. So I built a better number. Routed-expert volume counts only the weights that actually cross the bus, and you can get it from four fields in any GGUF header. Trinity spends 50% of its active budget on routed experts. Qwen spends 34%.

Then that one broke too.

On the 12GB 3060 at max offload, Trinity runs 23.70 tok/s against Gemma 4's 21.20 — and Trinity moves more routed bytes per token. My replacement rule ranks that pair backwards. What it ignores is the half of the model that never crosses the bus: Trinity's non-expert weights are roughly 36% lighter than Gemma's, and on a card that can't hold either model resident, that's enough to decide it.

What's left is two terms: bus traffic plus resident traffic. I can't give you a formula for it, because two rigs don't pin down the constants. So the pages say so, instead of pretending otherwise. Three of them now carry the correction rather than the claim.

📖 **The refined version, with both counterexamples printed next to it, is here.**

A New MoE, and Not From a Hyperscaler

Trinity-Mini is the model that broke both rules above: 26B total, ~3B active, Apache 2.0, from Arcee. I've now swept it on both rigs, the 3090 and the 12GB 3060, at every offload setting each card will take.

One thing stands out beyond the measurements. Every MoE I've put on these two rigs came out of a hyperscaler — Qwen from Alibaba, Gemma from Google. Trinity came out of a startup instead. That's the narrow version of the claim and it's the one I can defend: Arcee is VC-backed rather than independent in any romantic sense, and there's at least one publicly funded consortium MoE in this tier (Soofi S, 30B-A3B) I haven't put on a card.

It's not in the dataset yet. The rows need a second pass before they go in, and I'd rather they arrive late than arrive wrong.

Every Number Above Is Downloadable

The open benchmark dataset is still the place all of this ends up: two rigs, every flag, error bars, repetition counts, whether a display was attached, and the configs that failed to load. CC BY 4.0 — use it in a paper, a model card, or a spreadsheet. Attribution is the only ask.

📖 **The dataset, and the raw JSON, are here.**

That's the week. If you run a 24GB card and you've been offloading a couple of layers out of caution, that habit costs you 14% and buys you nothing. Go check what your card is actually holding.

New here, reading this on the web? Subscribe and the next one lands in your inbox.

— Mark, InsiderLLM

Measured something that contradicts one of these? Reply, or hit me at hello@insiderllm.com. I read everything — and this issue is what happens when I'm wrong.
