Qwen 3.8 isn't slow. It's just very, very thorough.

August 24, 2026 · by Mark Bartlett

Download this post as PDF

InsiderLLM Weekly issue 18 – August 24, 2026

14,953 tokens for one line of Python. Nine tokens for the identical line, from the identical file, with a single flag changed. The difference is the chat template, and it costs more than the model you picked.

Quick Hits

- **Hugging Face is reportedly exploring a sale at $13 billion or more.** Business Insider, 24 August: a bank is sounding out bidders, nothing is agreed, no buyer named. Reported, not confirmed — but worth knowing who might end up owning the place your weights come from.
- **Someone has quantised Qwen 3.8-27B down to a 16GB card.** IQ4_XS, 12.7-15.8 GB depending on which variant you pull, Apache 2.0, 1,525 downloads last month. I have not run it. Has anyone reading this?
- **There is now a DFlash2 draft model for 3.8.** 561 MB, Q2_K_S, 1,142 downloads — a draft for speculative decoding, not something you run on its own. Every DFlash number we have measured is on 3.6. 📖 Our DFlash numbers, on 3.6.

The Model Is Not Slow. It Generates 24 to 48 Times More Than You Asked For.

HumanEval problem 2 asks for the decimal part of a float. Give it 3.5, get back 0.5. On the chat path, at the model's own defaults, Qwen 3.8-27B spent **14,953 tokens** arriving at this:

```
return number - int(number)
```

The same file, same card, same seed, with the chat template switched off, writes that line in **nine tokens**.

Problem 108 is worse and stranger. Thirty-two thousand tokens, 14.2 minutes of an RTX 3090, and an empty response field — it hit the ceiling and returned no code at all. It had reasoned

coherently for most of it, then its generation degenerated into a single unbroken run of 30,037 zero characters. That is a decode collapse rather than a reasoning loop, and it is the only problem in all 164 that does anything like it. One problem, 7.2% of everything the model generated across the entire benchmark.

Across all 164: 444,556 tokens generated, 412,441 of them thinking. **Thinking is 92.8% of the output.** Against the raw-completion path, that is 24x more tokens at the median and 48x at the mean.

The number that changed how I think about context sizing is the spread. Answer length is bounded and boring — 0 to 602 tokens, median 173, exactly what a benchmark of small functions should look like. Thinking runs from 100 to 32,000 on one seed and 59 to 32,000 on another: **a 320x to 542x spread**, and answer length barely predicts it. Almost everything this model costs you lives in a variable you cannot see from the prompt, and a second seed widened the range rather than narrowing it.

So the obvious fix is wrong. More context does not help; 108 truncates at 16K and at 32K, and nothing about a collapse into repeated characters suggests 64K rescues it. The fix is a ceiling. `--reasoning-budget 8192` sits above the productive region and below the tail: **154 of the 164 problems never reach it** — on both seeds — and neither do 23 of the 25 that reasoning fixed.

Two traps in the knob, both found by dumping the template out of the GGUF and neither documented anywhere I can find. Setting `reasoning_effort` to `high` is silently rewritten to `xhigh`, so dialling back from the default quietly sets the default. And `medium` injects no instruction at all — it is the absence of steering, not moderate steering.

That 8,192 is fitted to this model. It held on both of 3.8's seeds and it is free on 3.6, but it is not a number to carry to a model I have not measured. The method transfers; the figure does not.

📖 **All 164 problems, the distribution, and where to put the cap, are here.**

The Protocol Was Worth More Than the Model

While that run was going I scored the completions, then did the same for 3.6, two seeds each. Four runs, 12.87 hours of GPU time on one 3090.

On the raw completion path these two models score 82.32% and 80.49% pass@1. Turn the chat template on and let them think, and they land at 95.12% and 95.73% for 3.6, 95.09% and 95.06% for 3.8.

Read those numbers twice. The gap **between the models** is 0.35 points. The gap between two seeds of the same model is 0.61. The comparison is not close — it is unresolvable at n=2, and paired McNemar returns p = 1.0000 on both seeds. Meanwhile the protocol change is worth **13 to 15 points, to both of them**.

That is a larger effect than anything separating a model generation, and it is entirely a decision you make at the command line.

Same total spend, opposite shape, and this is the part I would actually act on: 3.8 is bimodal — mean roughly 3x its median, cheap on most problems and occasionally catastrophic. 3.6 is uniform, mean and median nearly coincident, and across both runs it never once failed to terminate. Its thinking spans 12x to 29x against 3.8's 320x to 542x. All three truncations in the whole set belong to 3.8.

If you already run 3.6, none of this is a reason to move.

📖 **Speed, VRAM and the raw-path scores, with the conditions spelled out, are here.**

We Were Quoting a $1,000 3090. It Has Not Been $1,000 for Months.

A drift check caught a used 3090 priced at $1,000 across a page it had no business being on, and pulling that thread took 21 pages down with it.

The current pull, 68 sold listings: ordinary used cards cluster at a **$1,275 median**, eBay Refurbished sits separately at **$1,400**, and **nothing at all sold below $1,099**. Two distinct clusters, not one spread. The band on our pages is now $1,200-1,400.

The 5090 is worse. Against a $1,999 MSRP the August median sale was **$4,220** — **2.1x**, and that is what cards actually closed at, not what optimists list them for.

This changes advice rather than trivia. A used 3090 at $1,275 against a 5090 at $4,220 is roughly $53 per gigabyte against $132. Per gigabyte is not the only axis and the 5090 is substantially faster per token — I am not pretending otherwise — but it is the axis that decides whether a model loads at all, and fit is binary. Every "just buy the 5090" calculation on this site was carrying a 3090 price that stopped being true somewhere around May.

📖 **The full 3090 pull, venue by venue, with the red flags, is here.**

The $36 Fix

An old ThinkCentre M710q with one 8GB stick in a two-slot board. I wrote the prediction down before buying the RAM, which is the only reason this is worth reporting.

Second stick in, dual channel: token generation up **52-58%** across four models. Gemma 3 1B went 18.20 to 27.75 tok/s, Llama 3.2 3B 7.36 to 11.59. Prompt processing moved **under 2%**, which is the tell — generation is bandwidth-bound, prefill is not.

If you run anything on a CPU, go and look at how many slots are populated before you buy anything else.

📖 **Full guide here.**

That's the issue. If you are running Qwen 3.8 on the chat path right now, go and set `--reasoning-budget` before you do anything else — the default is uncapped, and the tail is where your evenings go.

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

— Mark, InsiderLLM

Run the 16GB IQ4_XS quant, or put DFlash2 on 3.8 before I did? Reply, or hit me at hello@insiderllm.com. I read everything.

Source: https://insiderllm.com/blog/newsletter-2026-08-24/