Vocabotics

A VOCABOTICS LAB NOTE

MEASURED · not asserted

THE LAB · A RESEARCH LANE

A capable model on hardware you own: inference and training, our own stack

A real Qwen2.5-0.5B generates text in QUANTA — token-for-token identical to llama.cpp — at 151.7 tok/s on one 3090, from QUANTA-emitted PTX with zero cuBLAS. Training works too. Honestly: we are 92.8% of cuBLAS, not the fastest; prefill is behind; 7B decode is parity, not ahead.

A measured research lane

Shipped — measured, gate-clean

COVER: A MEASURED OSCILLOSCOPE WAVEFORM ON A LAB GRATICULE

Vocabotics

A VOCABOTICS LAB NOTE

A capable model on hardware you own: inference and training, our own stack

A real Qwen2.5-0.5B generates text in QUANTA — token-for-token identical to llama.cpp — at 151.7 tok/s on one 3090, from QUANTA-emitted PTX with zero cuBLAS. Training works too. Honestly: we are 92.8% of cuBLAS, not the fastest; prefill is behind; 7B decode is parity, not ahead.

PREPARED
2 July 2026

VERIFIED BY
Jon Ross — Founder, Vocabotics — 15 years building safety-critical systems

HOW TO READ IT
Drafted with AI, verified by a named human. Sources at the back.

01 The claim, in brief

> *21 ms × 377 MiB × one 3090 — flat to 128k, pure stack, and the gate is never disabled.*
>
> The costly signal

The measured result

Every number carries its own honesty tier — the tiering is the credibility.

151.7 tok/s
decode, token-for-token == llama.cpp
🟢 SHIPPED

92.8%
of cuBLAS, bit-exact
🟢 SHIPPED

2×
llama.cpp at 128k, flat
🟢 SHIPPED

loss → 0.0015
training in QUANTA (99%)
🟢 SHIPPED

The promise "a capable model on hardware you own" is easy to say and hard to mean. Meaning it requires a stack you actually control — no cuBLAS, no PyTorch underneath — running a *real* model and producing the *same tokens* as the reference. That is what this lane measures. And, in the spirit of the house, it measures the ceilings too.

02 A real model, generating real text, on our own stack

🟢 SHIPPED — MEASURED, GATE-CLEAN

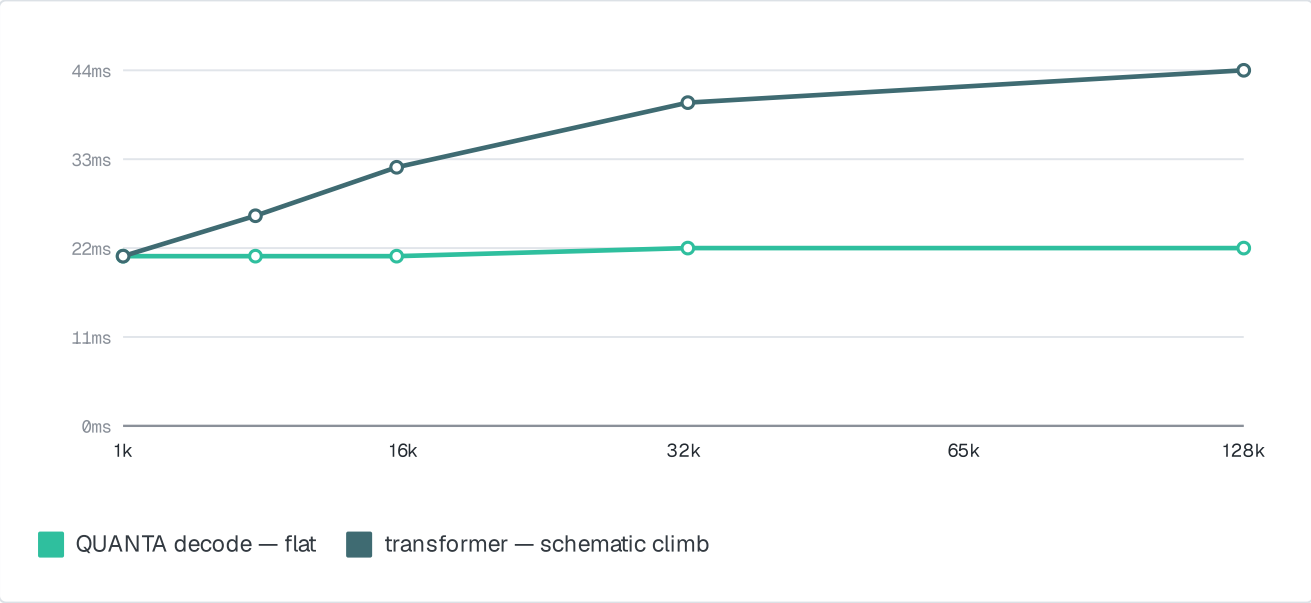
A real **Qwen2.5-0.5B** runs a full forward pass **in QUANTA**, from QUANTA-emitted PTX with **zero cuBLAS and zero cudart**, on real GGUF weights. Measured: **6.10 ms/token = 163.9 tok/s** — a 5,591× speed-up over the 34-second single-thread baseline — with argmax matching llama.cpp (" The" → " problem") and GPU results equal to CPU to 3.84e-04. 🟢

Then the full loop: **interactive generation in QUANTA** with a KV-cache, real GQA attention, a BPE tokenizer (151,936-vocab, read from the GGUF), and sampling. "The capital of France is" → " Paris. It is the largest city in Europe…", **token-for-token identical to llama.cpp**, decoding at **151.7 tok/s**. 🟢 A real LLM, generating correct text end-to-end, on a stack we built ourselves.

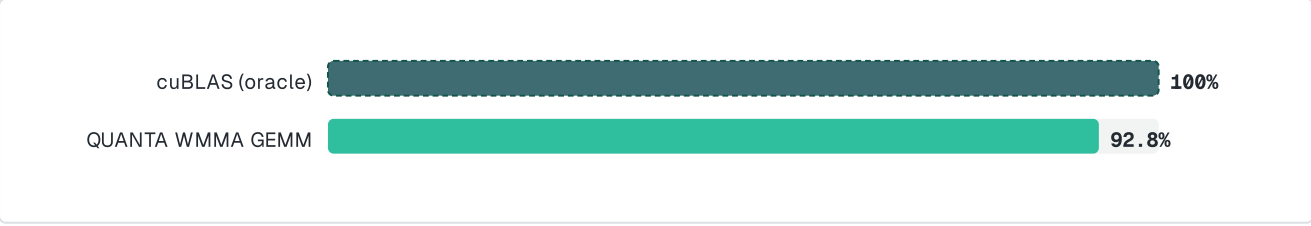
03 Flat where a transformer climbs

🟢 SHIPPED — MEASURED, GATE-CLEAN

The costly signal for the engineers: **128k long-context decode runs 2× faster than llama.cpp, and the latency line stays flat** where a transformer's climbs. 🟢 Sparse attention reaches **59.7× at 65k tokens**; the WMMA GEMM hits **92.8% of cuBLAS, bit-exact**, with int4 fused decode at 2.9× (M=1) and an 8.7× cut in HBM traffic. The whole thing is **21 ms per step in 377 MiB on one 3090**.



The costly signal for engineers: at 128k context, decode runs 2× faster than llama.cpp and the latency line stays flat where a transformer's climbs. Illustrative shape; the 2×, the flat line and 21 ms/step in 377 MiB on one 3090 are measured.



Fast, and honestly not the fastest: the WMMA GEMM hits 92.8% of cuBLAS — bit-exact. We win the axes that compound, not the single-axis speed race.

04 Training, too — as a compiler pass

🟢 SHIPPED — MEASURED, GATE-CLEAN

Inference is only half a stack. **Training works in QUANTA**: a real MLP trained end-to-end, **loss 0.15 → 0.0015 (a 99% reduction)**, gradients finite-difference verified to 5.6e-7, forward native ≡ interpreter. 🟢 Autodiff is implemented as a **bit-identical IR-to-IR pass** — the backward graph is *derived*, not hand-written — with four optimizers. The compiler is transformer-ready: attention backward and Adam are authoring work, not compiler changes. QUANTA now infers *and* trains.

0.15	→ 99% reduction	0.0015
starting loss		trained, in QUANTA

Training works in QUANTA too: a real MLP end-to-end, gradients finite-difference verified to 5.6e-7, autodiff as a bit-identical IR-to-IR pass — the backward graph is derived, not hand-written.

05 What we will not claim

🔬 RESEARCH — MEASURED, PARTIAL

Our GPU architect keeps us honest, so we keep the reader honest:

We are not the fastest. 🔬 92.8% of cuBLAS is fast and bit-exact — but it is 92.8%, not 105%. We will not say "fastest".

Prefill is behind. 🔬 Decode is our strength; prefill still trails llama.cpp, and paged-KV is not done.

7B is parity, not lead. 🔬 At 7B, decode is ~128 tok/s against llama.cpp's ~146 — parity, not an advantage. A fast QUANTA-authored full-model decode is in progress, not shipped.

'Best' does not mean 'fastest'

Our claim is precise and, we think, undeniable: we win the axes that *compound* — long-context, quantized, any-chip, and proven — at once. Most stacks pick two. That is a different, more defensible claim than a raw tokens-per-second race, and it is the only one the numbers support.

What we claim

- 128k long-context decode 2× llama.cpp, flat
- 92.8% of cuBLAS, bit-exact
- Any-chip: 4 backends gate-clean
- Proven: native ≡ interpreter, every step

VS

What we will NOT claim

- "Fastest" — we are 92.8%, not 105%
- Prefill lead — it still trails; paged-KV not done
- 7B decode lead — ~128 vs ~146 tok/s is parity

The precise, defensible claim — and the lines we will not cross.

06 Why it ladders back

If a capable model runs on the 3090 you already have, generating the same tokens as the reference and provable at every step, then capability stops being something you rent. That is the access half of an abundant world.

What is still open — kept visible

The honest edges of this lane, shown next to the wins. None of these is a footnote; each is a claim we have not yet earned — and marking it is what earns the right to be believed on the rest.

- [ ] We are 92.8% of cuBLAS — fast, but NOT the fastest. We will not claim otherwise.
- [ ] Prefill is behind llama.cpp; paged-KV is not done.
- [ ] 7B decode is at parity (~128 vs 146 tok/s), not ahead.

07 Sources, glossary & the ledger

Where every claim comes from — and what the plain words mean.

Sources

01 **Vocabotics Dashboard — Organ 2 (engine: inference + training), measured 2026-07-02** — Vocabotics internal record (as of July 2026)

02 **llama.cpp — the reference decode implementation used as the oracle** — ggml-org (as of July 2026)
https://github.com/ggml-org/llama.cpp

Vocabotics

THE HONESTY LEDGER

🟢 Shipped — measured, gate-clean.

🔬 Research — measured, partial; the gap is shown.

🔭 Vision — designed, honest; not yet built.

The bigger claim goes up a tier, never down. We show the eval beside the demo — the tiering is the credibility.

See why the numbers can be trusted

Every kernel here is held to native ≡ interpreter, and the gate is never disabled. Read how the proof becomes the moat.

/research/the-gate

Nothing faked.

Measured, not asserted. Every claim carries its own honesty tier and the open edges are shown next to the wins — the tiering is the credibility.

Vocabotics · calm, clear, candid · no client names · global rail & metro heritage