Preprint - measurement report, single-machine study. Not peer-reviewed. Prepared in IEEE conference layout for readability; no venue affiliation implied.

Chained Acceptance Is a Model Property: Field Measurements of Shipped Multi-Token-Prediction Heads for Depth-2 Speculative Decoding on a Memory-Bound GPU

Sebastian Haas

Independent researcher · kontakt@sebastianhaas.eu

***Abstract*-Several recent open-weight model families ship a trained multi-token-prediction (MTP) head alongside the backbone, enabling self-speculative decoding without an external draft model. Extending speculation from one drafted token to a chain of two is tempting on memory-bound hardware, but its value is governed by a quantity that is rarely reported for shipped heads: the *conditional acceptance of the second, chained draft*, α_2-the probability that the head's second guess survives verification given that its first one did. Across a controlled matrix of quantizations, an abliterated/base model pair, and two model generations with kernel-identical backbones, we find that α_2 behaves as a stable, trained property of the released model: on identical decode paths it is insensitive to weight quantization (72.7–74.2 % across three quantization recipes of the same model) and to abliteration-style fine-tuning, yet differs between model generations on structured output (72.5–74.2 % vs. 87–90 % on a fixed code benchmark; +8.6 points paired across a 12-prompt set, 7 of 8 code prompts positive), and it collapses to ≈49 % on free-form prose for *every* head we measured, successor included. A two-parameter cost model-expected tokens per step, $1+\alpha_1+\alpha_1\alpha_2$, against the measured cost ratio of the wider verification step-predicts the sign and magnitude of depth-2 gains within two percentage points across three architectures (dense recurrent hybrid, MoE hybrid with CPU-offloaded experts, and batched multi-stream serving), including two regimes where depth 2 *loses* despite high acceptance. Selecting the model by measured α_2 rather than adding kernels raised end-to-end single-stream throughput of a 27B dense hybrid from 45.8 to 50.3 tokens/s on a 2018-era 16 GB accelerator (1.55× over non-speculative decoding). We further show a consensus-gated lookup extension that appends a third, retrieval-based draft token only when it agrees with the head (+8.4 % on reproduction-heavy output, byte-identical results), and we document three measurement traps-text-path sensitivity of acceptance rates, prompt-set variance, and perplexity chunk-count mismatches-that we fell into ourselves and that can silently invalidate comparisons of this kind.**

***Index Terms*-speculative decoding, multi-token prediction, LLM inference, quantization, GPU serving, llama.cpp, acceptance rate, hybrid architectures, linear attention**

I. Introduction

Autoregressive decoding of large language models on consumer and end-of-life accelerators is memory-bandwidth-bound: every generated token streams the full working set of weights. Speculative decoding [1], [2] amortizes this stream by verifying several draft tokens in one forward pass. The past two years have shifted drafting from external models toward *trained heads* attached to the target model itself-Medusa [5], EAGLE [6], [13], and the multi-token-prediction (MTP) modules popularized by DeepSeek-V3 [4] and since shipped in open-weight releases such as the Qwen 3.x line [10]. For a growing number of GGUF releases, the draft head is simply *part of the download*.

This changes the practitioner's question. The classical literature asks how to train or choose a drafter; the operator of a shipped model asks: *given the head I already have, how deep should I speculate?* Depth 1 (one drafted token per step) is well understood and almost always positive on memory-bound hardware. Depth 2 verifies a *chain*: the head drafts token t+1 from the committed state, then drafts t+2 *from its own unverified guess*. Whether this pays depends on the conditional second-stage acceptance α_2-a quantity that model cards do not report, that differs from the first-stage rate α_1 in ways that are hard to guess, and that interacts with an architecture-dependent cost: on recurrent-hybrid backbones the extra verification row is *not* close to free, unlike on pure transformers.

This paper is a measurement study of exactly this decision, carried out end-to-end on a single fixed machine: an AMD Instinct MI50 (gfx906, 16 GB HBM2, capped at 150 W)-hardware that its vendor has retired from active ROCm support-driven by a performance-tuned fork of llama.cpp [9] and a purpose-built Rust serving layer. The constraint is deliberate: on a fixed, bandwidth-starved budget, algorithmic choices are exposed rather than masked by compute headroom.

We make four contributions:

1) *α_2 is a model property.* Across three quantization recipes of the same 27B model, an abliterated/base pair, and a same-architecture successor generation, second-stage chained acceptance moves by ≤1.5 points under quantization and abliteration (paired abliteration effect $+1.0 \pm 6.7$ points, null) but by 8–15 points on structured output between generations whose backbones we show to be kernel-identical (§IV). The shipped head-not the weights around it-determines depth-2 viability, and a two-minute probe measures it before any engineering is committed.

2) *A validated two-parameter decision rule.* Expected committed tokens per step, $E[T] = 1+\alpha_1+\alpha_1\alpha_2$, against the measured step-cost ratio $r = t_{d2}/t_{d1}$ predicts measured depth-2 outcomes within ≈2 points on three different serving regimes, including two instructive failures: a MoE hybrid with CPU-offloaded experts where depth 2 loses 19 % *despite* α_2 = 82–84 %, and saturated multi-stream batching where extra verification rows are pure added traffic (§V).

3) *Consensus-gated lookup extension.* A retrieval draft (prompt-lookup style [7], [8]) must not displace a trained head: displacement *lost* 5 % because copied text mis-continues incrementing patterns the head predicts correctly. Gated to fire only when retrieval agrees with both head drafts, the third token is nearly free: +8.4 % on reproduction-heavy output with byte-identical results (§VI).

4) *Negative results and measurement traps.* Acceptance rates are comparable only on identical text paths (a lm-head requantization alone shifted measured α_1 by 2 points); single-prompt acceptance figures scatter over ±15 points across prompts of the same task class; and perplexity is comparable only at matched evaluation-chunk counts-the same file scored 7.76 vs. 5.84 under a chunk-count mismatch we initially misread as a 3 % quantization loss (§III-C, §VII).

Everything here is measured on one machine, greedy decoding, with paired same-run comparisons wherever possible; §VIII states the limits of that scope explicitly. We release no new model; the point is a reusable decision procedure for heads that are already on practitioners' disks.

II. Background and Related Work

A. Speculative decoding and trained heads

Speculative decoding [1], [2] drafts k tokens cheaply and verifies them in one target-model pass; rejection sampling (or, under greedy decoding, exact prefix match) preserves the target distribution. Surveys [3], [14] cover the design space. Self-drafting attaches the drafter to the target: Medusa [5] adds independent per-offset heads, EAGLE [6], [13] an autoregressive feature-level head, and MTP training objectives [3], [4] yield heads that predict several future tokens. DeepSeek-V3's report [4] notes 85–90 % acceptance for its second predicted token; EAGLE reports per-position rates for its own trained drafters. What is *not* available in the literature, to our knowledge, is measured chained acceptance for the MTP heads that ship inside community GGUF releases, across the quantizations and fine-tunes those releases actually circulate in-the regime an end user operates in.

B. Retrieval drafting

Prompt-lookup decoding [7] and LLMA [8] draft by copying continuations of n-gram matches from the prompt/history-free drafts that work when output reproduces input. We use retrieval only as an *extension* of a trained head and show that the gating policy, not the retrieval itself, decides the sign of the gain (§VI).

C. Hybrid backbones make verification rows expensive

The models measured here are hybrids: Qwen 3.6/3.8-27B interleave gated-DeltaNet (GDN) linear-attention blocks [11] with full attention (48+16 of 64 blocks), and Nemotron 3.5 [12] is a Mamba-style MoE hybrid. Linear-recurrent blocks process positions *sequentially*; a verification batch of n_r rows cannot amortize the scan the way a transformer amortizes attention GEMMs. This raises the marginal cost of each added draft row and is, next to α_2, the second parameter of the depth decision. On our 27B dense hybrid the third row costs a measured factor r = 1.25 per step (§V-A); on the CPU-offloaded MoE the same widening costs r = 1.66 because every extra row re-streams expert weights over PCIe/DDR4.

D. Setting

All experiments run on one workstation: AMD Instinct MI50 (gfx906/Vega 20, 16 GB HBM2, ≈1.02 TB/s peak, power-capped at 150 W, fp16 26.5 TFLOPS), 32-core Threadripper 3970X host, DDR4. The software stack is a gfx906-tuned llama.cpp fork (Q4_0-centric kernels, fused MoE/GLU paths, graph-level MTP-head integration) plus a Rust serving layer implementing MTP streams, host-chained depth-2 drafting, continuous batching, consensus lookup, and recurrent-state prefix resume. At Q4_0 the 27B working set is 14.3 GiB; the roofline for non-speculative decode is ≈71 tokens/s, of which the measured baseline achieves 32.4 (45 %)-decode is firmly bandwidth-bound, so every accepted draft row amortizes one full weight stream.

III. Measurement Methodology

A. Probes

α_1 is counted per step as acceptance of the head's first draft under greedy verification. For α_2 we use two instruments. (i) A *probe* inside the measurement tool: after each committed step the head is fed its own draft, the resulting chained guess is recorded, and the probe row is rolled back from the head's KV state-α_2 becomes measurable without implementing depth 2 at all. (ii) The production depth-2 path (host-chained second head pass; verification batch of known+d1+d2), whose accounting reports accepted2/drafted2, i.e., acceptance of d2 *conditional* on d1 acceptance. Both instruments agree on the model ranking; production numbers are used in all tables unless marked.

B. Controls

Comparisons are paired: same prompt, greedy decoding, warm model, cooling gate (≤44 °C edge before each run), fixed 150 W cap, and-after we caught the DVFS trap in §VII-default power-management state. Output hashes are recorded; any A/B claim marked *byte-identical* produced bit-equal text. Perplexity anchors use WikiText-2 at matched chunk counts within the same run (§III-C). Backbone equivalence between the two model generations is established with speculation disabled: tg64 32.28 vs. 32.48 tokens/s, pp512 271.6 vs. 271.6 -identical within run noise, so *all* serving differences

reported later are attributable to the head and the decode policy.

C. Traps that invalidate naive comparisons

Three effects, each encountered the hard way, bound what may be compared with what:

1) *Text-path sensitivity of α.* Any change that perturbs logits-even requantizing only the lm-head-changes the generated text, hence the drafting substrate, hence measured acceptance: the identical head scored α_1 = 88.6 % vs. 90.7 % purely across an output-tensor requantization. α comparisons are valid only between runs that generate the same token path, or must be aggregated over prompt sets.

2) *Prompt-set variance.* On a 12-prompt set (§IV-C), single-prompt α_1 ranges from 66 to 96 % *within the code class*. Any single-prompt acceptance number-including several in our own earlier lab notes-carries ±15-point uncertainty about the class mean.

3) *Perplexity chunking.* PPL is comparable only at equal evaluation-chunk counts: the same file scored 7.76 (16 chunks) vs. 5.84 (32 chunks) on the same corpus. We therefore report only same-run pairs (identical chunking) and label the chunk count (PPL16/PPL32).

IV. A_2 Is a Model Property

A. Invariance under quantization and abliteration

Table I fixes the decode stack and prompt and varies only the released file: three quantization recipes of the 3.8-generation 27B (IQ4_NL, Q3_K_M, imatrix-Q4_0 with Q4_0 output/embedding requant "B2"), its abliterated variant, and the successor generation Qwen3.6-27B under the identical B2 recipe. α_1 wobbles with the text path (§III-C); α_2 stays inside 1.5 points across every 3.8 variant-then jumps 13–16 points for the 3.6 head on an unchanged backbone (§III-B; pooled over the prompt set the jump is +8–9 points, §IV-C). Abliteration, which measurably costs language quality (Table V), does not touch the head: the abliterated variant has the *best* α_1 of its family on the B2 text path.

TABLE I
Chained acceptance across releases of one backbone (single structured code prompt, greedy, identical stack; production d2 accounting except † = probe; prompt-set statistics in Table II)

Released file (27B class)	α_1 (%)	α_2 (%)
Qwen3.8 base, IQ4_NL	86.2	72.7
Qwen3.8 base, Q3_K_M †	88.1	73.0
Qwen3.8 base, B2-Q4_0	93.2	73.5
Qwen3.8 abliterated, i1-Q4_0	88.6	73.6
Qwen3.8 abliterated, B2-Q4_0	90.7	72.5–74.2
Qwen3.6, B2-Q4_0 (same arch)	**94.4**	**87.1–90.1**

The practical reading: nothing an end user does to the file -quant choice, requant recipe, abliteration-moves α_2 materially. What moves it is which head the vendor trained. A separately shipped head GGUF for the 3.8 generation is bit-wise the same head; transplanting it is pointless, and we abandoned that route after this table.

B. Generation gap and its signature

The 3.8 head loses little on its first guess (α_1 88–93 %, comparable to 3.6's 94–96 %) but 14–17 points on the chained one. The drop from α_1 to α_2 is −19 points for 3.8 vs. −6 for 3.6 on the fixed benchmark prompt, and −14.2 vs. −9.8 pooled over the code prompt set (§IV-C)-milder, same order. This asymmetry is the signature of the training recipe: a head trained only to predict position t+2 from *ground-truth* context degrades when fed its own guess, while a head trained on its own chained rollouts does not. We cannot inspect the vendors' training code, but the measurement cleanly separates the two hypotheses quant-vs-training, and it transfers: a Mamba-MoE hybrid's shipped head scored α_2 = 84.2 % (probe), again a fixed property of the release.

C. Prompt-set statistics

Single-prompt acceptance values are seductive and unstable (§III-C). We therefore reran the three B2-Q4_0 releases-abliterated 3.8, base 3.8, and 3.6-over a 12-prompt set (8 code/structured, 4 free prose; 200 tokens each, greedy, production depth-2 path), ≈550 chained drafts per model and class. Table II and Fig. 1 give the statistics; three results survive aggregation.

1) *The abliteration effect is null at scale:* paired per prompt, the abliterated model differs from its base by +1.0 ± 6.7 points α_2 on code (4/8 prompts positive) and +0.3 ± 12.2 on prose-no effect, now with n ≈ 550 rather than one prompt.

2) *The generation gap is real but task-bound:* +8.6 ± 13.0 points paired on code (7/8 prompts positive; pooled 79.6 % vs. 71.5/70.4 %), yet *zero* on prose (−0.8 points paired). The successor's head is a structured-output specialist, not a uniformly better chainer-and the fixed benchmark prompt of Table I (+15 points) sits at the favorable end of the per-prompt spread.

3) *Within-class variance is large:* per-prompt α_1 spans 66–96 % and α_2 spans 57–93 % *inside the code class*. Any depth decision computed from one prompt-including several early ones in our own lab notes-can be ±10 points off the class mean; probes should average a small set.

TABLE II
Prompt-set acceptance statistics (12 prompts, 200 tokens each, greedy, production d2; mean ± SD over prompts, pooled = Σaccepted/Σdrafted)

Release (B2-Q4_0)	Class	α_1 (%)	α_2 (%)	α_2 pool
3.8 abliterated	code	85.9 ± 5.3	71.6 ± 8.8	71.5
3.8 base	code	87.0 ± 6.4	70.6 ± 11.3	70.4
3.6	code	**89.6 ± 4.4**	**79.7 ± 6.8**	**79.6**
3.8 abliterated	prose	72.7 ± 5.9	49.9 ± 8.0	49.7
3.8 base	prose	75.9 ± 3.8	49.5 ± 8.2	49.5
3.6	prose	73.8 ± 5.9	48.9 ± 15.9	48.4

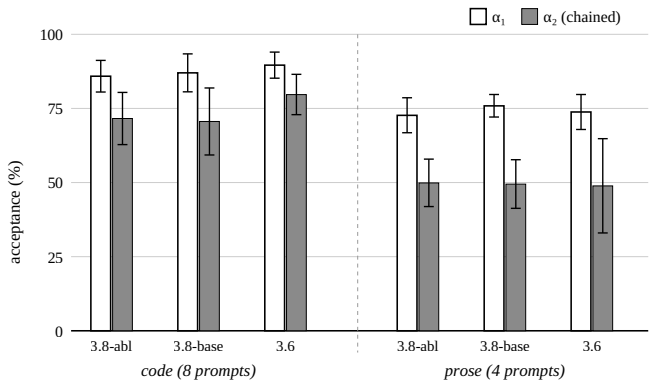

Fig. 1. First-stage (α_1, white) and chained (α_2, gray) acceptance over the 12-prompt set; bars are per-prompt means, whiskers ±1 SD. The three shipped heads are indistinguishable on prose and separate only on structured output, where the successor (3.6) head chains ≈8–9 points better; the abliterated/base pair is statistically identical everywhere.

D. Task dependence

Chained acceptance is strongly task-dependent in a way first-stage acceptance only hints at. On free-form prose α_2 collapses to ≈49 % pooled for all three heads (Table II)-single prompts go as low as 31–44 %-while α_1 only drops to ≈73–76 %. The same collapse appears on a 9B sibling (85.0 % → 42.3 % code→prose) and on the Q3 quant (84.0 % → 44.1 %), so it is neither a scale nor a quantization artifact. The naive assumption $\alpha_2 = \alpha_1$ would misprice the chained term by a factor of ≈1.5 on prose and flip the deployment decision (§V).

V. The Economics of Depth 2

A. A two-parameter rule

Under greedy verification, a depth-2 step commits

$$E[T] = 1 + \alpha_1 + \alpha_1\alpha_2 \quad (1)$$

tokens versus $1+\alpha_1$ at depth 1, at step cost t_{d2} versus t_{d1}. Depth 2 wins iff

$$(1+\alpha_1+\alpha_1\alpha_2) / (1+\alpha_1') > r, \quad r = t_{d2}/t_{d1}, \quad (2)$$

where α_1' is the depth-1 arm's own acceptance. Both sides are measurable in minutes: the left with the §III-A probe, the right from one step timing. The rule is deliberately free of hardware modeling; r *is* the hardware model.

B. Three regimes, decomposition vs. measurements

Two claims must be kept apart. *Ex ante*, the §III-A probe plus one step timing decides the build: it placed code above and prose below break-even on the dense hybrid, and the offloaded MoE below, before any depth-2 code existed-correct side in all four regimes, though its absolute gain estimate is text-path-limited to a few points (§III-C). *Post hoc*, Table III tests whether (2) is *complete*: with the committed-tokens ratio taken from step counts and r from independently instrumented step timings, the model reproduces the measured wall-clock ratio to within ≈0.3 points on the dense hybrid and the MoE-there are no hidden costs beyond the two parameters.

The regimes span the interesting space. On the dense GDN hybrid the third verification row costs r = 54.5/43.5 ms = 1.25 -the linear scan processes rows sequentially, so the row is far from free. On code, the gain factor 1.29 clears it narrowly (+2.8 %); on prose the same hardware ratio meets a collapsed α_2 and loses. On the MoE hybrid with CPU-offloaded experts, every extra row re-streams expert weights over host memory: r = 1.66, and depth 2 loses 19 % *despite* α_2 = 82 %-high acceptance is necessary, not sufficient. In saturated 3-stream batching (GPU ≈97 % busy) extra rows are pure added traffic and depth 2 loses ≈25 %. Speculation depth is therefore a per-regime setting, not a per-model constant.

TABLE III
Decomposition of eq. (2) against end-to-end measurements (greedy, paired runs; E-ratio from step counts, r from step timings)

Regime	α_1/α_2 (%)	E_{d2}/E_{d1}	r	eq. (2)	meas.
27B dense, code	89.7 / 72.5	1.288	1.253	+2.8 %	+2.8 %
27B dense, prose	66.3 / 42.7	1.114	1.25*	−10.9 %	−7.8 %
30B-A3B MoE, experts on CPU	93.7 / 82.2	1.354	1.662	−18.5 %	−18.8 %
27B, 3-stream batch (saturated)	89.2 / 75.2	-	-†	<0	≈−25 %
27B successor head (3.6), code	94.4 / 87.1	1.42‡	1.25*	+14 %	+9.8 %‡

*Code-load row timing reused; not separately timed on this load-the residual is consistent with a slightly cheaper prose/3.6 row. †At 97 % GPU occupancy added rows serialize; no meaningful per-row timing. ‡Cross-model comparison at equal backbone speed (§III-B), E-ratio from acceptance rates: 50.3 vs. 45.8 t/s end-to-end; the residual is second-order (step-count-dependent fixed costs).

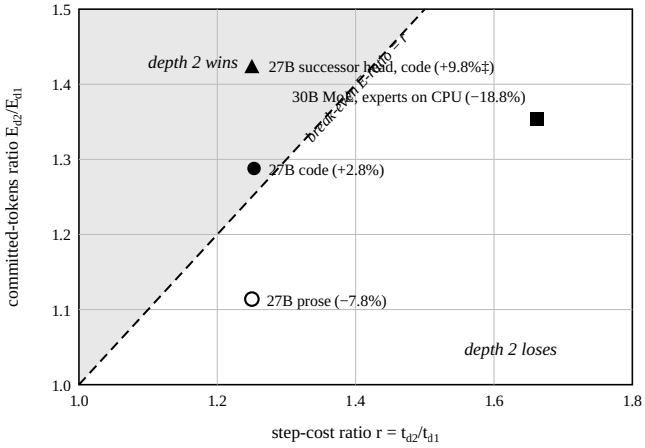

Fig. 2. The depth-2 decision plane of eq. (2). Each point is one serving regime, placed by its measured committed-tokens ratio (y) and step-cost ratio (x); the shaded half-plane above the diagonal is where depth 2 pays. High chained acceptance alone does not decide the outcome: the MoE point sits far below break-even at α_2 = 82 % because CPU-offloaded experts make every verification row expensive, while the same α_2 ≈ 73 % that barely clears break-even on code falls below it on prose. Annotations give measured end-to-end deltas (‡ cross-model, Table III).

C. What this bought end to end

Table IV summarizes the serving outcomes on the fixed 16 GB card. The single largest lever was not a kernel: replacing the 3.8-generation release by its successor-selected purely

because the probe showed $\alpha_2 \approx 90$ on an identical backbone-moved single-stream code throughput from 45.8 to 50.3 tokens/s (1.55× over the 32.4 t/s non-speculative baseline). The requant recipe that makes the file fully GPU-resident (lm-head and embeddings to Q4_0, 15.3 → 14.3 GiB) costs +0.5–1.2 % same-run perplexity and returned +29 % throughput on the 3.6 release-the cheapest tokens/s in the whole campaign.

TABLE IV
End-to-end single-stream serving, 16 GB MI50, greedy, Q4_0-class files

Configuration (27B dense hybrid)	tok/s	× base
no speculation (either generation)	32.4	1.00
3.8-abl.: depth 1	44.6	1.38
3.8-abl.: depth 2 (code)	45.8	1.41
3.8-abl.: depth 2 + lookup, reproduction load	58.1	1.79
3.6: depth 2 (code)	**50.3**	**1.55**
3.6: depth 2 + lookup, reproduction load	**59.3**	**1.83**
3-stream aggregate, depth 1 (3.8-abl.)	60.7	-

VI. Consensus-Gated Lookup Extension

Reproduction-heavy output (rewriting files, echoing command lists) invites retrieval drafting [7], [8]: continue the most recent history match of the current 6-token suffix. Our first version *displaced* the head's drafts with the retrieved chain when a match existed-and lost 5 % (50.9 vs. 53.5 t/s) while acceptance fell. The failure is systematic, not noise: copied text continues incrementing patterns incorrectly (`checkpoint_001` is followed by `001` again, where the ground truth increments to `002`), precisely the positions where the trained head is right. On reproduction load the head is nearly perfect on its own ($\alpha_1 = \alpha_2 = 99.0$ %); retrieval must not outrank it.

Version 2 inverts the authority: the head's two drafts stand; only if the retrieved chain *agrees with both* is its next token appended as a third draft (two independent signals concurring). Fired this way, d3 almost always survives verification: steps for 200 tokens drop 101 → 89, +8.4 % throughput (53.5 → 58.1 t/s), byte-identical output, and on non-reproduction load the gate simply stays closed (+0.5 %, no regression). The same mechanism with the head disabled provides lookup-only speculation for releases without heads: +4 % on code for a 20B MoE at 127 tok/s, byte-identical.

VII. Negative Results and Traps

We record what failed, since each failure is a reusable warning:

1) *In-graph chaining is not automatically faster.* Moving the second head pass (and even the full head) into the main compute graph-eliminating per-step flush round-trips-was throughput-neutral to negative (44.6–45.8 host-chained vs. 44.5–44.7 in-graph): the flush time was GPU work, not submission overhead, and tail-drafting from a rejected branch forfeits the next step's draft (Markov effect, $E[T] = 2/(2-\alpha) < 1+\alpha$).

2) *Depth 2 in saturated batching is traffic, not latency hiding* (Table III, row 4).

3) *Single-request verification hides lifecycle bugs.* A host-buffer pinning leak crashed every *second* request; months of single-request A/B runs were structurally blind to it. Serving claims need multi-request soaks.

4) *Accounting indices are part of correctness.* A drafts-per-step anomaly (drafted = steps − accepted) sat in every measurement for a week; the cause was an API that indexes logits by batch row, silently returning "no draft" after accepted steps. The fix alone was worth +7 % single-stream-more than most kernels we built. Instrument the invariant (drafted = steps) rather than the average.

5) *DVFS state is an experimental variable.* A manually raised GPU performance level, left over from an unrelated test, produced sawtooth throughput at the 150 W cap and invalidated a day of absolute numbers; paired A/B ratios survived. Cooling gates and power-state resets belong in the run protocol, not in folklore.

Finally, the quality side of the abliteration pair (Table V): abliteration costs a measurable but small +1.2–1.9 % perplexity at matched quantization, and-per Table I-leaves the head intact. The successor generation dominates the class: better PPL than both 3.8 variants *and* the fastest serving path.

TABLE V
Quality anchors (WikiText-2, PPL32, same-run chunking) vs. serving speed

File (27B class)	PPL32	tok/s (d2)
3.8-abl. Q4_K_M (quality quant)	5.5997	(not resident)
3.6 B2-Q4_0	**5.6553**	**50.3**
3.8 base Q4_0	5.6942	38.0
3.8 base B2-Q4_0	5.7722	46.0
3.8-abl. i1-Q4_0	5.8041	42.0
3.8-abl. B2-Q4_0	5.8404	45.8

VIII. Limitations

This is a single-machine study: one GPU family (gfx906), one inference stack, greedy decoding only. α under sampling temperatures >0 will differ (typed acceptance replaces exact match). The model matrix covers one backbone family in depth (three quants, one fine-tune pair, one successor) plus two hybrids as outside points; broader coverage-more vendors' heads, more hardware, temperature sweeps-is needed before the invariance claim can be called general. The prompt set (12 prompts, two task classes) bounds within-class variance but is small, and the headline single-prompt values (Tables I, IV) sit at the favorable end of the per-prompt distribution; perplexity anchors are WikiText-2 only and say nothing about task accuracy. The cost model (2) ignores second-order step-count effects on fixed costs (visible as the 4-point residual in Table III, row 5). All numbers are for shipped GGUF releases as users download them; we did not retrain or fine-tune any head.

IX. Conclusion

For shipped MTP heads, the depth-2 decision reduces to two cheaply measurable numbers: the chained acceptance α_2-a stable property of the release, indifferent to quantization and

abliteration, strongly dependent on the vendor's head training and on the task-and the step-cost ratio r, a property of architecture and placement (recurrent scans and CPU-offloaded experts make verification rows expensive; saturated batching makes them pure cost). Equation (2) with these two inputs predicted every sign and magnitude we measured within ≈2 points. The practical protocol is: probe α_2 before building anything; requantize for residency (it is nearly free); select the release by its head; speculate at depth 2 only where (2) clears; extend by retrieval only under consensus gating. On retired 16 GB hardware this protocol-not new kernels-delivered the last 10 % to 50.3 tokens/s single-stream on a 27B dense hybrid.

Acknowledgment

Experiments, analysis, and manuscript were prepared with substantial AI assistance (Claude, Anthropic) operating the author's workstation under the author's direction; all measurements ran on the author's hardware and are reproducible from the project's benchmark logs. The gfx906 kernel work builds on the llama.cpp project [9] and its community.

References

[1] Y. Leviathan, M. Kalman, and Y. Matias, "Fast inference from transformers via speculative decoding," in *Proc. ICML*, 2023.

[2] C. Chen, S. Borgeaud, G. Irving, J.-B. Lespiau, L. Sifre, and J. Jumper, "Accelerating large language model decoding with speculative sampling," arXiv:2302.01318, 2023.

[3] F. Gloeckle, B. Youbi Idrissi, B. Rozière, D. Lopez-Paz, and G. Synnaeve, "Better & faster large language models via multi-token prediction," in *Proc. ICML*, 2024.

[4] DeepSeek-AI, "DeepSeek-V3 technical report," arXiv:2412.19437, 2024.

[5] T. Cai, Y. Li, Z. Geng, H. Peng, J. D. Lee, D. Chen, and T. Dao, "Medusa: Simple LLM inference acceleration framework with multiple decoding heads," in *Proc. ICML*, 2024.

[6] Y. Li, F. Wei, C. Zhang, and H. Zhang, "EAGLE: Speculative sampling requires rethinking feature uncertainty," in *Proc. ICML*, 2024.

[7] A. Saxena, "Prompt lookup decoding," 2023. [Online]. Available: https://github.com/apoorvumang/prompt-lookup-decoding

[8] N. Yang et al., "Inference with reference: Lossless acceleration of large language models," arXiv:2304.04487, 2023.

[9] G. Gerganov and contributors, "llama.cpp: LLM inference in C/C++." [Online]. Available: https://github.com/ggml-org/llama.cpp

[10] Qwen Team, "Qwen3 technical report," arXiv:2505.09388, 2025, and subsequent Qwen 3.x releases with NextN/MTP modules.

[11] S. Yang, J. Kautz, and A. Hatamizadeh, "Gated delta networks: Improving Mamba2 with delta rule," in *Proc. ICLR*, 2025.

[12] NVIDIA, "Nemotron-H: A family of accurate and efficient hybrid Mamba-transformer models," arXiv:2504.03624, 2025, and subsequent Nemotron hybrid releases.

[13] Y. Li, F. Wei, C. Zhang, and H. Zhang, "EAGLE-3: Scaling up inference acceleration of large language models via training-time test," arXiv: 2503.01840, 2025.

[14] H. Xia et al., "Unlocking efficiency in large language model inference: A comprehensive survey of speculative decoding," in *Findings of ACL*, 2024.