WHEN DOES ACCEPTANCE PREDICTION HELP ADAPTIVE SPECULATIVE DECODING?

Anonymous authors
Paper under double-blind review

ABSTRACT

Adaptive speculative decoding uses serving-time signals to choose how much to draft or verify. When do these signals identify useful budget changes, and when does that knowledge improve throughput? We study these questions through per-round counterfactual replay and controlled serving experiments. In the evaluated block-diffusion lattices away from draft-window saturation, confidence predicts accepted length but weakly distinguishes rounds that benefit from one additional beam width. Frozen learned allocators change accepted length by -0.4 to $+0.5\%$ at matched node budgets, despite hindsight gains of 6–11%. A failure–repair decomposition localizes this gap: the evaluated signals predict whether verification fails much better than whether a wider tree repairs that failure. Positive results at smaller budgets, saturated windows, and with an official EAGLE-3 head delimit this finding. For verification depth, the tested survival predictors yield useful allocation gains. With the decision and learner held fixed, target-side features available before drafting recover approximately half of the post-draft learner's allocation gain on three homogeneous text benchmarks; discarded drafts also retain useful signal after prefix correction. Separate live draft-horizon experiments expose two further constraints: execution cost and the static baseline. In one controlled stream at concurrency one, a target-state rule improves the measured token rate by 1.33% over the best global constant but loses to a frozen workload-specific constant. These measurements distinguish acceptance prediction, actionable budget information, and realized serving benefit. They support decision-specific evaluation rather than a general limit on adaptive decoding.

1 INTRODUCTION

Speculative decoding accelerates generation by verifying several cheaply proposed tokens in one target-model pass (Leviathan et al., 2023; Chen et al., 2023). Its efficiency depends on how much computation is spent drafting and verifying candidates. Recent methods adapt this computation using draft confidence, acceptance history, learned predictors, and hardware cost models (Li et al., 2024; Wu et al., 2025; Zhang et al., 2024; 2026; Lin et al., 2026; Cheng et al., 2026; Liu et al., 2026b). A central measurement question remains: *when does predicting acceptance identify a beneficial change in the decoding budget?*

The answer depends on the decision. A request whose current tree accepts a long prefix need not benefit from a wider tree. Low confidence may identify a likely failure without identifying an additional branch that repairs it. Verification depth has a different structure: another position is useful when the accepted prefix reaches it. Evaluating a signal only by how well it predicts accepted length can therefore obscure the quantity required by a particular allocator. Even useful allocation information need not improve throughput if obtaining or acting on it costs more than the resulting computation saves.

Prior work optimizes draft-token selection across requests (Wu et al., 2025), derives marginal acceptance–cost criteria (Lin et al., 2026), and studies why better predictors need not yield better policies for lossy verifier skipping (Luo et al., 2026). We examine a different setting: budgets for candidates that remain target-verified. Our central finding is a contrast within that setting. Local widening often has little predictable repair value in the tested lattices, whereas earlier target

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Table 1: The decision determines the counterfactual and the evaluation. Earlier signals may also be used to choose post-draft controls.

Control	Commit point	What changes?	Evidence used here
Horizon γ	Before drafting	Generated candidates	Live throughput; truncation only approximates a different horizon
Width W / nodes n	After drafting	Selected candidate tree	Per-round replay; executed-width validation on 27B text
Verify depth k	After drafting	Prefix of existing drafts	Per-round replay; a separate live prototype on the 4B system

states and even discarded drafts support useful verification-depth allocation. Execution controls and calibrated static choices then test how modest gains survive deployment.

We separate draft horizon, verification width, and verification depth. The horizon changes the candidates generated; width and depth select among candidates already produced. Recorded lattices support counterfactual accepted-length measurements for the latter two at fixed round starts, with replayed width labels checked against execution on the primary text system. Live horizon experiments then measure execution costs and baseline sensitivity. They are deployment case studies, not an end-to-end validation of the different verification-depth task.

Findings. Weak repair prediction accompanies small local-width allocation gains. At matched budgets, frozen learned allocators obtain -0.4 to $+0.5\%$ accepted-length change in the lower-saturation configurations, beneath hindsight gains of approximately 6–11%. The signals distinguish failure of the current tree much better than repair by the wider tree. We report the positive cases alongside this result: saturated windows, small budgets, and coarser node-budget decisions, including an official EAGLE-3 head (§3).

Useful verification-depth information survives before the next draft. Holding the allocator and learner fixed, target-side features recover approximately half of the post-draft learner's gain on MBPP, GSM8K, and MATH-500. The previous lattice also helps: its discarded tail adds predictive value beyond the previous acceptance outcome, including under a conditional-shuffling control. These are achievable gains of tested predictors, not information-theoretic bounds (§4).

Execution cost and static heterogeneity can absorb modest live gains. Fixed-action controls measure an approximately 5% observation loss on our overlapped stack. On a controlled stream at concurrency one, a target-state horizon rule gains 1.33% in R over the best global constant, but loses against a workload-specific constant frozen on calibration data. Other streams and an incomplete prospective boundary test limit the generality of this comparison (§5).

The practical contribution is a decision-specific evaluation procedure: measure the relevant budget increment, separate within-request candidate selection from cross-request allocation, and test the resulting policy against both execution controls and a calibrated static alternative.

2 Decisions and Measurement Protocol

Three controls, different counterfactuals. A block-diffusion drafter fills $\gamma - 1$ masked positions in one forward pass, anchored by the last committed token. In our primary drafter, each position has $K = 16$ token candidates and a selector scores transitions between consecutive positions. These candidates and scores form a lattice. A width-W beam produces a prefix-closed verification tree. The target accepts a drafts and commits one additional token, giving accepted length $\tau = a + 1$. We distinguish the controls in Table 1; c denotes client concurrency.

For a round r and two budgets b, b', define the acceptance level, gain, and positive-gain event:

$$L_r(b) = \tau_r(b), \qquad \Delta\tau_r(b, b') = \tau_r(b') - \tau_r(b), \qquad E_r(b, b') = \mathbf{1}\{\Delta\tau_r(b, b') > 0\}. \tag{1}$$

AUROC for E_r measures discrimination of a particular event, not expected gain or throughput. The learned width allocator also regresses gain magnitudes; it is not restricted to a binary classifier. Offline policy gains are relative changes in summed τ at a matched budget. An oracle uses the realized acceptance curves at those same starts. It is a hindsight comparator, not a prediction of deployable throughput; its action set must also be stated.

Systems and data. The primary target is Qwen3.8-27B, a hybrid gated-delta-net/attention model, with its released block-diffusion drafter (Chen et al., 2026; Inco AI, 2026); the second is full-attention Qwen3-4B with a drafter of the same family. We additionally study Muse-Glimmer-30B and two EAGLE-3 heads: a public Qwen3-4B checkpoint and the authors' Llama-3.1-8B-Instruct checkpoint (Li et al., 2025). Experiments use bf16 on B200 GPUs and a fork of SGLang (Zheng et al., 2024), with thinking disabled unless stated. The video data contain questions from MMBench-Video, Video-MME, and MVBench. The initial video population has 16,418 request-rounds from 203 requests; its full trace has 169,611 rounds. Text collections use HumanEval, MBPP, GSM8K, MATH-500, and MT-Bench, with 480 requests per block-diffusion system. Appendix A specifies collections, splits, and learning procedures.

What replay identifies. Under greedy verification, the accepted path matches a prefix of the target continuation. We can therefore score an alternative tree or truncation against the recorded lattice and continuation without another target pass. Replay reproduces 169,610 of 169,611 video request-rounds and all chain rounds. On 1,414 teacher-forced 27B text starts, executed and replayed width events agree in 99.3–99.9% of cases ($\kappa = 0.94$–0.99); the residual arises from bf16 changes with verification shape (Appendix B). Sampling experiments use *sample matching*, not the stack's default rejection sampler. Coupled target draws permit replay under this rule; we do not claim the measured marginal-event AUROCs transfer to other verifiers.

Replay keeps round starts fixed. Changing acceptance changes the next start, history features, and potentially batch composition. Nor does truncating a deep draft reproduce a shallow one: executed short horizons exceed the truncation estimate by 0.31 and 0.19 tokens at $\gamma = 8$ and 12 in a calibration experiment. All throughput results below are measured live.

Splits and uncertainty. Frozen width policies are selected on fitting requests and evaluated on disjoint requests; video training excludes any request sharing its first 16 output tokens with a test request. Without this exclusion, a learned width predictor's AUROC rises from 0.55 to 0.64. Offline intervals resample requests on text and prompt fingerprints on video. Since video identity was not logged, different questions on the same video may remain dependent. For live comparisons we use the request-level token rate $R = \sum_q N_q / \sum_q \ell_q$, where N_q is output tokens and ℓ_q is request latency; gains are $100(R_{\text{policy}}/R_{\text{base}} - 1)$. At concurrency above one, overlapping request latencies make R different from wall-clock system tokens/s. We label the latter separately. Live intervals combine prompt and launch variation. We retain the original family of 43 live comparisons for multiplicity adjustment, including comparisons no longer emphasized in this presentation. Two-launch results remain sensitive to a weakly estimated launch component; Appendix G reports the small-sample sensitivity.

Relation to existing policies. EAGLE-2 and related tree methods adapt which candidates a request verifies (Li et al., 2024; Wang et al., 2025); TETRIS and Bole also allocate verification resources across requests (Wu et al., 2025; Wang et al., 2026). LibraSpec explicitly optimizes marginal acceptance against verification cost (Lin et al., 2026). BlockPilot selects a block size once after prefill, while AdaEAGLE learns a draft-length predictor (Zhang et al., 2026; 2024). DSpark and D-cut address verification and its serving implementation (Cheng et al., 2026; Liu et al., 2026b); AdaFlash combines a length head with on-policy drafter updates (Qian et al., 2026). Confidence-guided local repair (Liu et al., 2026a) and lossy verifier skipping (Luo et al., 2026) are also close precedents. We distinguish their candidate-generation and verifier-skipping actions from fixed-candidate allocation, without claiming a same-hardware ranking of the complete systems (Appendix H).

3 Width: Predicting Failure versus Predicting Repair

Allocation headroom need not be predictable. On the video trace, an exact dynamic program reallocating widths at a matched budget obtains $+12.7$, $+10.6$, and $+7.4\%$ accepted length relative to uniform $W = 2, 4, 6$. Yet confidence separates long and short accepted lengths much better than it separates rounds helped by an additional unit of width. At $W = 4$, its AUROC is 0.71 for the level and 0.48 $[0.46, 0.49]$ for the event $a(5) > a(4)$. A level target with comparable prevalence reaches 0.85, so event sparsity alone does not explain the difference (Appendix C).

Table 2: **Local width and node-budget allocation are different tasks.** Panel A reports frozen learned-allocation gains at a matched node budget (DP for beam rows), with request/fingerprint-clustered 95% intervals. AUROC diagnoses the local $4 \to 5$ event, not the allocator's complete gain curve; text saturation is measured at $W = 4$. Panel B shows positive cases without equating their protocols: the official EAGLE-3 row predicts budget doubling and uses five-fold evaluation. Node-budget oracles there allow all integer budgets. These are offline gains, not throughput.

System / window / control	Saturation	Learned AUROC	Allocator gain (%)	Oracle (%)
A. Lower-saturation lattice settings; greedy unless marked				
27B video, 7 drafts, $W = 4$	.12	.55	-0.2 $[-0.6, +0.2]$	$+10.6$
27B video, 7 drafts, $W = 6$	.14	.55	-0.3 $[-0.7, +0.1]$	$+7.4$
27B video, best-first, 28 nodes	—	—	$+0.5$ $[+0.2, +0.8]$	$+9.7$
27B text, 15 drafts, $W = 4$	.17	.55	$+0.2$ $[-0.1, +0.4]$	$+6.9$
sample matching	.14	.55	$+0.0$ $[-0.3, +0.3]$	$+7.3$
4B text, 7 drafts, $W = 4$	.27	.57	$+0.4$ $[+0.3, +0.6]$	$+8.5$
sample matching	.25	.57	$+0.2$ $[+0.0, +0.4]$	$+8.8$
4B text, 15 drafts, $W = 4$	.01	.52	-0.1 $[-0.4, +0.1]$	$+9.6$
sample matching	.01	.53	-0.4 $[-0.7, -0.2]$	$+9.7$
B. Positive cases: window saturation, smaller width, and a different action				
27B text, 7 drafts, $W = 4$	.53	.65	$+1.06$ $[+0.77, +1.33]$	$+5.2$
27B text, 15 drafts, $W = 2$	.17[†]	—	$+0.66$ $[+0.27, +1.05]$	$+8.4$
Official EAGLE-3 / Llama-8B, 31 nodes	—	.722	$+1.16$ $[+1.08, +1.24]$	$+6.5$

[†]The saturation entry describes the trace at $W = 4$, not the $W = 2$ action.

The operational consequence is small allocation gain, not literal absence of signal. Table 2 shows frozen learned policies on video and both text targets, including sample-matching decoding. In its lower-saturation block, gains range from -0.4 to $+0.5\%$; several small gains exclude zero. The learned dynamic program predicts accepted length at $W = 1$ and each subsequent increment, then optimizes their cumulative curves under the budget. Thus the result also concerns a magnitude-aware allocator, not only the binary-event probe. The beam rows fix that learner; the best-first row uses a separately frozen learned node allocator. Selection among the beam learner and four rule-based alternatives is reported in Appendix C.

Failure and repair require different predictions. Let T_W be a prefix-closed tree and y the fixed target continuation. If it fails inside a window of P drafts, its first failure is at $d = a(W) + 1$. Define

$$F_1 = \mathbf{1}\{a(W) < P\}, \qquad F_3 = \mathbf{1}\{y_{1:d} \in \mathrm{Pref}_d(T_{W+1})\}, \qquad E_W = F_1 F_3, \tag{2}$$

setting $F_3 = 0$ when $F_1 = 0$. This identity follows from prefix closure: a wider tree accepts more precisely when it contains the continuation through the old failure position. It does not require nesting between widths. Set-based replay verifies its implementation on 181,281 rounds; the empirical finding is the difference in predictability of the two factors.

At the evaluated deployed widths, confidence predicts F_1 with AUROC 0.87–0.97, while the tested statistics discriminate $F_3 \mid F_1$ weakly (0.51–0.58); learned probes reach 0.53–0.57 (Figure 1). The realized token is present in the candidate pool in 63–82% of these failure rounds across confidence terciles. Hence pool absence alone does not explain the weak repair prediction. Even revealing the failure position and probing the drafter's local scores helps little at $W = 4 \to 5$. At $W = 1 \to 2$, however, that diagnostic probe reaches 0.65–0.68. The failure position is privileged information; this is evidence of a boundary, not a deployable predictor.

Boundaries of the result. Three comparisons matter. First, when many rounds already fill the draft window, predicting saturation identifies requests that cannot benefit from widening. On the 27B seven-draft trace, 53% of rounds saturate, and confidence predicts the marginal event in reverse (AUROC 0.31). The 0.4 trace-level saturation cutoff separating the original groups was chosen after examining the data; it is not an identified transition. Second, smaller budgets provide positive gains on both text systems and the video's best-first family. Third, on the official EAGLE-3 head, *doubling a node budget* is predictable: learned AUROC is 0.705–0.722, with gains of $+1.16$–$+2.54\%$ across three budgets. The weak EAGLE-3 checkpoint is therefore not representative of the family. Confidence-guided local repair can also change which branches are generated, as in CURE (Liu et al., 2026a); our fixed-candidate width increment does not test that intervention.

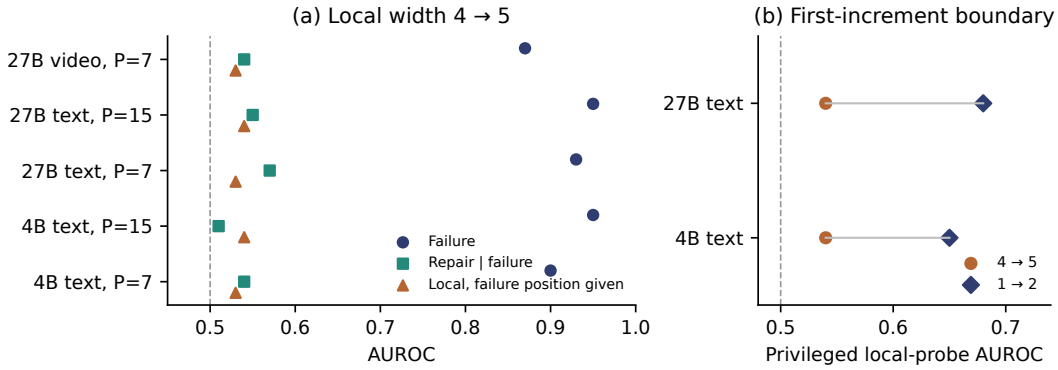


Figure 1: **Failure is easier to predict than repair in the local-width experiment.** AUROCs of chain confidence for F_1 and $F_3 \mid F_1$, and of local scores given the failure position, from the same decomposition table (Appendix C). Points are reported estimates, not confidence intervals. The right panel isolates the privileged local probe's stronger performance at the first width increment. Neither panel includes node-budget doubling.

The tree construction baseline also matters. On the frozen video evaluation, temperature-flattened best-first selection *within each request* improves accepted length by 4.88% over the deployed beam at the same node total; the selected cross-request allocator yields a further 0.50% $[0.19, 0.78]$ relative to that baseline (the best-first row of Table 2). On the weak EAGLE-3 head, improved within-request ordering similarly leaves 0.2–0.5% for cross-request allocation. A gain attributed to adaptive allocation must be separated from a better candidate ordering.

Across 24 node-budget cells, margin AUROC and the fraction of an integer-budget oracle recovered co-vary. Because those cells have only five drafter/target clusters, change budget together with predictability, and mix natural and synthetic batches, we treat the relation as exploratory (Appendix E). We do not infer an AUROC threshold or an upper bound on recoverable gain. Menu and unrestricted oracles are distinguished there.

4 Verification Depth: What Is Known before Drafting?

Verification depth has a simpler marginal event. For a full draft with accepted prefix a, verifying only k drafts gives

$$\tau(k) = 1 + \min(a, k), \qquad \tau(k) - \tau(k-1) = \mathbf{1}\{a \geq k\}. \tag{3}$$

Thus a survival predictor directly scores an additional verification slot. We hold the full draft, allocation budget, learner, and data splits fixed and change only its input features. The allocator assigns $\sum_i k_i = nG$ slots using predicted survival probabilities, preserving prefix closure. The video allocator requires $k_i \geq 1$ and uses $G = 4$ of 7 drafts; the text allocator allows $k_i = 0$ and uses $G = 7$ of 15. Text batches randomly pool eight recorded request-rounds, rather than reconstructing co-resident requests. These are accepted-length gains over uniform truncation, not throughput estimates.

Discarded drafts retain useful signal. On video, learned history gives $+0.15\%$, adding the previous outcome gives $+0.62\%$, and adding the whole previous lattice gives $+1.98\%$ $[+1.02, +2.69]$. Most of the increment comes from the *stale tail*: positions beyond the previous acceptance point that overlap the next draft, despite its corrected prefix. The tail adds 1.09 points $[0.42, 1.63]$ over the outcome rung, versus 0.43 $[0.04, 0.72]$ for the front.

A conditional-shuffling control strengthens this interpretation. In a separate fitted run, replacing each tail by another round's tail with the same previous accepted length, in both training and test, reduces the tail rung from $+1.76\%$ to $+0.61\%$, near the outcome-only $+0.58\%$. The shuffled-minus-true difference is -1.15 points $[-1.61, -0.54]$. Masking more of the tail also progressively removes its benefit. On the measured EAGLE-3 chain, stale drafts contribute much less (Appendix D); we do not infer that this is universal to autoregressive drafters.

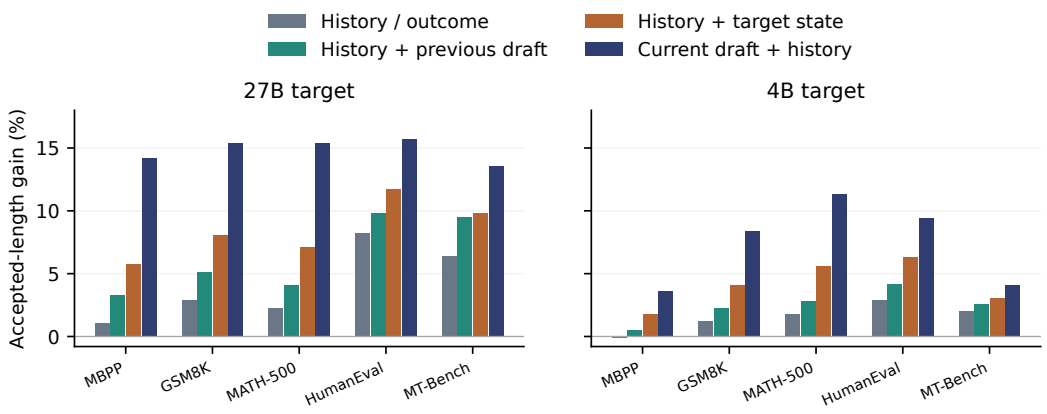


Figure 2: **Pre-draft signals have useful, benchmark-dependent allocation value.** Accepted-length gain over uniform verify depth, with the same full draft and budget, from the re-collected text traces. The first three predictors use information available before the current draft. The previous-draft and target-state inputs each extend the history/outcome baseline separately. The current-draft comparator uses lattice features and history, without the separate outcome column or target features. Values are point estimates; pooled and ratio intervals are in Appendix D. No predictor execution cost is included.

Target-side features provide the strongest tested pre-draft input. We next add logits from the previous verification pass and a 256-dimensional projection of the target state at its last accepted position. This builds on the use of target states for length prediction (Zhang et al., 2024). Figure 2 reports absolute gains by benchmark. Averaged over MBPP, GSM8K, and MATH-500, this input recovers 0.46 $[0.38, 0.52]$ and 0.50 $[0.45, 0.53]$ of the post-draft learner's gain on the 27B and 4B systems; the previous lattice recovers 0.28 and 0.22. HumanEval and MT-Bench have larger pre-draft fractions and are displayed rather than folded into that three-benchmark summary. Sample-matching results preserve the ordering (Appendix D).

Information must be priced at its available age. An overlapping host reads features with a delay. With inputs two rounds old, the pooled 27B target-feature rung falls from $+13.2\%$ to $+10.2\%$, while history alone gives $+7.2\%$. The comparison measures the value retained by these predictors at a longer feature delay; it does not bound all information available in the earlier state.

History also partly distinguishes workloads. On the original 27B pooled trace, a trailing mean gains 7.2%, compared with 0.5–1.6% on MBPP, GSM8K, and MATH-500 separately. A hindsight depth fixed per benchmark already gives 5.4% on that pooled budget. This motivates a workload-aware baseline for live policies. It does not establish that heterogeneity is the sole source of history's value: within-request persistence and between-request variation co-vary in the measured cells.

5 Realized Value: Execution Cost and Static Baselines

The offline study establishes allocation value for existing candidates. Deployment additionally requires an inexpensive execution path and a meaningful baseline. We examine these issues with live *horizon* policies on the primary 27B system. This changes the action from §4; the live predictors are lightweight heads rather than its gradient-boosted learners. We therefore use the experiments to identify concrete deployment effects, not to attribute every offline-to-live difference to signal timing.

A controlled horizon experiment. We use 120 calibration and 400 disjoint held-out requests from Dolly chat and GSM8K-train, native CUDA graphs, equal admission, and horizons $\{8, 12, 16\}$. Policies use acceptance history, a target-state head, or post-draft confidence; a cost-aware rule selects using predicted accepted tokens and measured round time. Fixed-action arms run the adaptive machinery without changing the horizon, and an observational arm computes, reads, and logs the signals without acting on them. Each arm has two launches on rotated GPUs.

Table 3: **Policy gains and execution controls answer different questions.** 400 held-out 27B requests, two launches per arm. Policy gains are relative to the best global constant at each load ($\gamma = 12$ at $c = 1$, 8 at $c = 8$). Controls compare against a native constant at the *same* horizon. All entries are request-rate R changes (%) with two-level 95% intervals, not simultaneous confidence intervals.

Arm	$c = 1$	$c = 8$
History, no new synchronization	$+1.00\ [0.25, 1.74]$	$-2.35\ [-3.46, -1.23]$
Target state, no new synchronization	$+1.33\ [0.67, 1.99]$	$-1.96\ [-2.85, -1.08]$
Post-draft confidence, host read	$-3.81\ [-4.49, -3.12]$	$-6.68\ [-7.92, -5.45]$
Cost-aware rule	$+0.54\ [-0.18, 1.26]$	$-1.78\ [-3.07, -0.48]$
Machinery, action at best constant	$-0.92\ [-1.20, -0.64]$	$-0.94\ [-1.35, -0.54]$
Observe/read/log, fixed $\gamma = 16$	$-5.32\ [-5.68, -4.97]$	$-4.10\ [-4.47, -3.74]$

Table 4: **Workload-specific static choice is a stronger baseline on this stream.** The assignment is frozen before evaluation. Reconstruction uses the original constant-arm records; the served confirmation is a later measurement wave. Each gain uses its corresponding comparator records, so the two columns must not be cross-subtracted. No batched reconstruction is claimed. The later served design also changes the engine arrangement.

Request-rate comparison ($c = 1$)	Reconstructed	Served confirmation
Workload lookup / best global constant	$+2.23\ [1.55, 2.90]\%$	$+2.69\ [2.03, 3.35]\%$
Target-state rule / workload lookup	$-0.88\ [-1.50, -0.25]\%$	$-1.92\ [-2.52, -1.32]\%$

Table 3 separates policy gains from fixed-action overhead. The target-state rule gains 1.33% in R $[0.67, 1.99]$ over the best global constant at $c = 1$; history gains 1.00% $[0.25, 1.74]$. Under Holm adjustment over the retained 43-test family, the former survives ($p = 0.0019$) and the latter does not ($p = 0.14$). A Satterthwaite correction for the small launch count widens the target-state interval to $[0.65, 2.00]$ and gives adjusted $p = 0.0046$, preserving that conclusion. Every horizon policy in this experiment has a negative gain in R at $c = 8$.

A read has a measurable cost on this path. At a fixed $\gamma = 16$, the post-draft signal predicts accepted length slightly better than target state: $R^2 = 0.289$ versus 0.257 at $c = 1$. However, observing that signal stalls the host–device pipeline. A later, two-launch decomposition holds accepted length fixed at $\tau = 4.5836$ and measures round times of 16.928, 16.922, 17.668, and 17.893 ms for the native constant, constant-action machinery, read-only control, and full observational arm. The read alone adds 0.746 ms and reduces throughput by 4.23% $[3.75, 4.70]$; head computation and logging contribute a further 1.26% relative to the read-only arm. The components multiply to the full 5.40% loss, rather than adding independent charges.

This control identifies an implementation cost. It does not establish an unavoidable tax on post-draft signals: DSpark uses delayed capacity estimates to hide scheduling latency, and D-cut uses packed verification with graph buckets while describing an overlap limitation (Cheng et al., 2026; Liu et al., 2026b). Nor does the read explain losses of policies that never synchronize. At $\gamma = 16$ the machinery-only effect is indistinguishable from zero in the later wave; we do not infer horizon-dependent amortization from a cross-wave comparison.

A stronger static baseline changes the conclusion. An adaptive policy can exploit differences *between* workloads without making useful round-level decisions within each one. We therefore freeze a separate constant on calibration prompts: $\gamma = 8$ for chat and 12 for mathematics. At $c = 1$, the resulting lookup is evaluated both by selecting the appropriate executed per-request constant-arm records and by serving a workload-routed pair of engines (Table 4). The reconstruction requires request cost to be independent of neighboring requests, a premise checked on this stream; it is not used under batching.

Both estimates favor the lookup, but their difference is material relative to these small gains. Measured arm drift between waves reaches 0.73%, and the two-engine arrangement is an additional deployment difference. The evidence is strongest for the direction on this stream, not a precise de-

composition of the gap. In the served comparison, charging the lookup the 0.92–1.04% machinery overhead still leaves the rule approximately 0.9–1.0% behind. The lookup assumes a free workload label; a learned label-free per-request static selector, as motivated by BlockPilot (Zhang et al., 2026), is not tested here.

What transfers, and what remains unresolved. Other live streams support the importance of the baseline but not universal static dominance. A history rule gains 1.61% $[1.03, 2.18]$ on a stream with more predictable acceptance; on 400 fresh mixed requests it gains 0.76% $[-0.01, 1.53]$, which does not survive multiplicity correction. Across seven recorded streams, its comparison with a per-workload constant has no interval strictly above zero, but some point estimates are positive (Appendix F).

Prospective tests on three systems predicted signs more successfully than magnitudes. A sharper test of the workload account—whether the rule's margin over a frozen lookup changes as specified between two new streams—remains incomplete after failed launches. We retain that unresolved primary prediction rather than replacing it with the easier observation that identical static choices have zero difference (Appendix I).

Finally, the primary stack's recurrent-state layout does not support per-request verification depth. A live prototype on the full-attention 4B system does, but runs in eager mode. At $c = 8$, history loses 5.1% in R, target state loses 17.0%, and post-draft confidence ties the best uniform block ($+0.4\%$ $[-0.6, 1.4]$). These results document a limited implementation, not the achievable online value of the information measured in §4.

6 Discussion and Conclusion

The measurements suggest three checks before attributing a gain to adaptive speculation. **Match the prediction target to the decision:** acceptance level, a positive marginal event, and expected marginal gain are distinct quantities. **Match the comparator to the policy:** separate within-request tree improvements from cross-request allocation, and report an oracle with the same feasible actions. **Price the implementation against a strong static alternative:** control signal observation and machinery costs, and distinguish workload-level choice from round-level adaptation.

Our strongest result is conditional. On the evaluated block-diffusion lattices, the tested learners obtain small local-width gains alongside weak repair prediction. Verification depth admits useful pre-draft prediction, including from discarded candidates. Positive width and node-budget results delimit that finding. Live horizon experiments show that observation costs and static workload differences can absorb modest gains, without establishing that every adaptive policy must lose.

Limitations. The study uses one serving stack and hardware generation. Local replay holds starts fixed; sampling coverage is limited to sample matching; actual width execution was checked on 27B greedy text. Video identity is missing, synthetic batches enter several offline comparisons, and the saturation grouping was chosen after observing the traces. Learned probes give achievable performance, not information-theoretic limits. Small launch counts and cross-wave drift constrain the precision of live gains. A graph-compatible, same-allocator verification-depth experiment and the incomplete prospective workload test would be needed for a stronger bridge between the offline and online findings.

Reproducibility statement

Appendices give replay definitions, learning protocols, full retained tables, and the status of prospective tests. The supplementary archive contains analysis scripts, small result files, serving patches, and recorded predictions. A separate minimal evidence package contains saved survival predictions and a standalone CPU check of Figure 2. Full serving traces and the model environment are not included in that minimal package; the historical scripts also require path configuration. Appendix J distinguishes these materials. Invalidated runs and protocol deviations are recorded separately.

AI USE STATEMENT

Generative AI tools assisted manuscript editing, literature discovery, code development and debugging, feedback on experimental methodology, interpretation and consistency checks of recorded results, and figure preparation. The authors take responsibility for the final manuscript, code, and scientific claims.

REFERENCES

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*, 2023.

Jian Chen, Yesheng Liang, and Zhijian Liu. DFlash: Block diffusion for flash speculative decoding. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2602.06036; code at `https://github.com/z-lab/dflash`.

Xin Cheng, Xingkai Yu, Chenze Shao, Jiashi Li, Yunfan Xiong, Yi Qian, Jiaqi Zhu, Shirong Ma, Xiaokang Zhang, Jiasheng Ye, Qinyu Chen, Chengqi Deng, Jiping Yu, Damai Dai, Zhengyan Zhang, Yixuan Wei, Yixuan Tan, Wenkai Yang, Runxin Xu, Yu Wu, Zhean Xu, Xuanyu Wang, Muyang Chen, Rui Tian, Xiao Bi, Zhewen Hao, Shaoyuan Chen, Huanqi Cao, Wentao Zhang, Anyi Xu, Huishuai Zhang, Dongyan Zhao, and Wenfeng Liang. DSpark: Confidence-scheduled speculative decoding with semi-autoregressive generation. *arXiv preprint arXiv:2607.05147*, 2026.

Inco AI. Qwen3.8-27B-DFlash2 model card. `https://huggingface.co/incoai/Qwen3.8-27B-DFlash2`, 2026. Vendor-published accepted lengths and throughput. Accessed 2026-09-23. Mirrored at `https://huggingface.co/z-lab/Qwen3.8-27B-DFlash2`.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning (ICML)*, 2023. arXiv:2211.17192.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. EAGLE-2: Faster inference of language models with dynamic draft trees. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 7421–7432, 2024. doi: 10.18653/v1/2024.emnlp-main.422. arXiv:2406.16858.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. EAGLE-3: Scaling up inference acceleration of large language models via training-time test. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. arXiv:2503.01840.

Zexun Lin, Yuan Feng, Junlin Lv, Kevin S. Zhou, and Xike Xie. LibraSpec: Dynamic diffusion-based speculative decoding via marginal-gain-driven optimization. *arXiv preprint arXiv:2608.08721*, 2026.

Aofan Liu, Jingxiang Meng, Fangxin Liu, and Yongbiao Chen. CURE: Local uncertainty repair for block-parallel speculative decoding. *arXiv preprint arXiv:2608.00531*, 2026a.

Tianyu Liu, Yuhao Shen, Rui Cen, Junhan Shi, Jiebin Zhang, Guangshuo Qin, Hong Liu, Song Liu, Guanghua Yu, and Jianchen Zhu. D-cut: Adaptive verification depth pruning for batched speculative decoding. *arXiv preprint arXiv:2607.14647*, 2026b.

Haoxuan Luo, Jameson Sandler, and Ferdinando Fioretto. From positionwise confidence to prefix scheduling: Verifier skipping in speculative decoding. *arXiv preprint arXiv:2608.14787*, 2026.

Yu-Yang Qian, Hao-Cong Wu, Chen Chen, Jiacheng Sun, Zhenhua Dong, Peng Zhao, and Zhi-Hua Zhou. AdaFlash: Adaptive speculative decoding via on-policy distilled diffusion drafters. *arXiv preprint arXiv:2607.19223*, 2026.

Jikai Wang, Yi Su, Juntao Li, Qingrong Xia, Zi Ye, Xinyu Duan, Zhefeng Wang, and Min Zhang. OPT-Tree: Speculative decoding with adaptive draft tree structure. *Transactions of the Association for Computational Linguistics*, 13:188–199, 2025. doi: 10.1162/tacl_a_00735. arXiv:2406.17276.

Li Wang, Yi Su, Xiabao Wu, Chiran You, Yongchao Liu, Zhan Qiu, Juelu Zhang, Jiajun Zheng, Fangxin Liu, Jie Zhang, Chen Tian, and Chengying Huan. Bole: Efficient tree speculation for hybrid-attention language models. *arXiv preprint arXiv:2608.01651*, 2026.

Zhaoxuan Wu, Zijian Zhou, Arun Verma, Alok Prakash, Daniela Rus, and Bryan Kian Hsiang Low. TETRIS: Optimal draft token selection for batch speculative decoding. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 33329–33345, 2025.

Hao Zhang, Yiming Hu, Yong Wang, Mingqiao Mo, Xin Xiao, and Xiangxiang Chu. BlockPilot: Instance-adaptive policy learning for diffusion-based speculative decoding. *arXiv preprint arXiv:2606.31315*, 2026.

Situo Zhang, Hankun Wang, Da Ma, Zichen Zhu, Lu Chen, Kunyao Lan, and Kai Yu. AdaEAGLE: Optimizing speculative decoding via explicit modeling of adaptive draft structures. *arXiv preprint arXiv:2412.18910*, 2024.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark Barrett, and Ying Sheng. SGLang: Efficient execution of structured language model programs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

A COLLECTIONS, FEATURES, AND LEARNING PROTOCOLS

Collections and units. The initial video collection contains 203 requests in 140 prompt-fingerprint clusters; the full collection contains 2,027 requests in 634 clusters and 169,611 request-rounds. Fingerprints use the first 16 output tokens. They reduce repeated-prompt leakage but do not identify the source video. For text, each system uses 100 prompts each from HumanEval, MBPP, GSM8K, and MATH-500, and 80 from MT-Bench, with up to 512 new tokens at concurrency four. The original 27B and 4B collections contain 20,874 and 36,324 chain-valid rounds. Complete 15-token windows retain 19,931 and 34,496 rounds; complete seven-token windows retain 20,348 and 35,529. The separately collected target-feature traces retain 19,962 and 34,415 complete 15-token windows, from 478 and 480 retained requests, respectively. Counts from these collections should not be interchanged.

The text depth and frozen-width scripts randomly permute recorded request-rounds and group eight at a time. They do not require distinct requests or matched progress within a group. In the pooled target-feature depth analysis, 10.02% of the 27B groups and 7.39% of the 4B groups contain multiple rounds from one request. These groups define a fixed-start resource allocation problem, not feasible co-resident serving batches. Other replay collections use their recorded or separately constructed groups. A separate 200-request, concurrency-eight natural-batch check contains 1,171 multi-request batches and 9,031 rounds (mean batch size 7.7). Its depth oracle, confidence policy, and history policy yield approximately 28.0, 21.2 $[18.0, 24.3]$, and 8.1 $[5.3, 11.2]\%$. This check supports their ordering in that collection, not numerical transfer of every synthetic-batch result.

Benchmark sources. The public sources are linked below. These identify the benchmarks, not the exact local snapshot or sampled request IDs; those are properties of the collection scripts. Video questions were used as generation workloads rather than to reproduce their official accuracy leaderboards. The MT-Bench collections here use first-turn requests.

Benchmark	Primary source
HumanEval	Evaluating Large Language Models Trained on Code
MBPP	Program Synthesis with Large Language Models
GSM8K	Training Verifiers to Solve Math Word Problems
MATH-500	HuggingFaceH4 dataset card
MT-Bench	Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena
MMBench-Video	OpenCompass dataset card
Video-MME	Video-MME benchmark paper
MVBench	MVBench benchmark paper

Features and learners. History contains earlier acceptance outcomes and request progress. Lattice features summarize candidate probabilities, margins, and selector scores; previous-lattice features are aligned relative to the preceding acceptance point. Target logit features summarize top-1 log probability, top-1/top-2 margin, and entropy estimated from the top 32 logits at each of 16 verification positions, including a residual-tail approximation. They also summarize the preceding acceptance position and accepted prefix. The hidden feature is a fixed 256-dimensional projection plus one log norm of a concatenation of five layers at the last accepted position (257 dimensions in total). Current-lattice features are available only after the current draft. Using a feature from a completed target pass does not by itself make that feature free to read on the host.

Gradient-boosted predictors use LightGBM with row subsampling 0.8, feature subsampling 0.5–0.6, and L_2 regularization 5. Video depth models use 600 trees, 63 leaves, learning rate 0.05, and minimum leaf size 100. Text depth models use 300 trees, 31 leaves, learning rate 0.06, and minimum leaf size 50; text width diagnostic models use 400 trees and learning rate 0.05. The frozen text width gain regressors instead use 300 trees, 31 leaves, learning rate 0.05, minimum leaf size 50, and feature subsampling 0.5. Video width models use up to 2,000 trees, 63 leaves, learning rate 0.03, and early stopping on 10% of training requests. Logistic controls standardize features and use $C = 0.1$. MLP controls have hidden sizes 2048 and 512, GELU activations, dropout 0.2, AdamW with learning rates 10^{-3} or 3×10^{-4}, weight decay 0.05, batch size 4096, and at most 12 epochs, selected by validation AUROC.

Frozen width evaluations split fitting and evaluation requests in half. Video predictors train on two other servers, excluding evaluation fingerprints. Other text prediction diagnostics use five request folds; they are labeled separately from frozen policy selection. Round-level random splits are not used as evidence of generalization to new requests.

B Replay Definitions and Validation

Greedy prefix replay. Let $\mathrm{Pref}_j(T)$ be the set of length-j token prefixes in a prefix-closed tree, and let y be the target's greedy continuation from the recorded start. Then

$$a(T; y) = \max\{j : y_{1:j} \in \mathrm{Pref}_j(T)\}, \qquad \tau(T; y) = a(T; y) + 1. \tag{4}$$

For truncation to depth k, prefix closure gives $\tau_k = \min(a, k) + 1$. For a change from T to T', if $a(T) < P$, improvement occurs exactly when $y_{1:a(T)+1} \in \mathrm{Pref}_{a(T)+1}(T')$. If $a(T) = P$, improvement inside the same window is impossible. This proves Eq. 2, including when the two trees are not nested. Non-nesting can also cause negative gains. The identity is a property of the replay definition, not a statistical discovery about a drafter.

This argument assumes the same target continuation. Exact arithmetic under causal masking supplies it for greedy verification, whereas bf16 execution can change logits with tensor shape. The executed-width validation in Table 5 quantifies that discrepancy on 27B greedy text. The one disagreement in the full video's replay check similarly prevents an assertion of bitwise exactness. Neither check validates complete counterfactual trajectories.

Table 5: Executed versus replayed width labels on 1,414 teacher-forced 27B text starts. Each hypothetical width is replayed from the width-4 server's lattice and continuation. Agreement and Cohen's κ compare $a(W') > a(W)$ with its executed label. Starts were sampled along the output rather than at naturally occurring rejection boundaries.

drafts, increment	executed prevalence	agreement	κ	executed a decreases
$7, 1 \to 2$	17.5%	99.7%	0.99	0.2%
$7, 4 \to 5$	3.8%	99.9%	0.98	0.2%
$7, 5 \to 6$	2.6%	99.7%	0.94	0.1%
$15, 1 \to 2$	30.7%	99.3%	0.98	0.1%
$15, 4 \to 5$	8.5%	99.5%	0.97	0.4%
$15, 5 \to 8$	15.0%	99.4%	0.98	0.4%

Sample matching. At each accepted prefix, sample a fresh token from the target conditional distribution. If it matches the selected draft token, commit it and continue; otherwise commit that target token and end the round. Inductively, each committed token has the target conditional law. The draft selector must not anticipate the fresh target draw. Coupling those draws permits fixed-start counterfactual replay under this verifier. This is a different construction from acceptance-ratio rejection sampling, so its predictability results are verifier-specific. Table 6 compares accepted lengths only: sample-matching/default ratios range approximately 0.97–1.10 on 27B and 0.92–0.98 on 4B. Similar means would not establish similar marginal-event labels.

Table 6: Chain accepted length under greedy, sample-matching, and default rejection-sampling verification. Both sampling rules use temperature 1, top-p 0.95 and top-k 20; 100 prompts per benchmark, 80 for MT-Bench, concurrency 4, single runs. Similar mean lengths do not establish similar marginal-event predictability.

	27B			4B		
	greedy	sample match	rejection	greedy	sample match	rejection
HumanEval	10.84	10.74	10.46	4.72	4.56	4.18
MBPP	8.19	7.23	7.44	3.98	3.87	3.57
GSM8K	7.91	7.15	7.29	5.08	4.76	4.56
MATH-500	8.08	7.42	7.60	5.59	5.16	5.02
MT-Bench	4.42	3.80	4.17	3.13	2.97	2.90

C Width Allocation and Failure–Repair Controls

Magnitude-aware allocation. For widths 1 through 12, separate regressors estimate $a(1)$ and the subsequent increments. Negative predicted increments are clipped to zero and accumulated into a predicted curve. An exact discrete dynamic program chooses widths with the specified total budget; its realized score uses the actual replayed acceptance curve. The hindsight comparator optimizes that actual curve. Actual curves are not assumed concave or perfectly monotone: non-monotonicity occurs in fewer than 0.24% of natural starts and about 0.4% of teacher-forced starts. Consequently, a greedy marginal selector is not interchangeable with the exact oracle.

Four rule-based alternatives and learned DP enter the separately frozen arm-selection protocol. Selection uses fitting requests only. Table 7 retains every window/width cell, including gains outside the lower-saturation subset; Table 8 distinguishes the populations used for hand statistics and leak-free learned probes. The level AUROC uses $a(4) \geq \mathrm{median}(a(4))$; the sparsity-matched control uses $a(4) \geq 7$, and the depth event is $a \geq 5$. Pearson correlation uses the corresponding unbinarized quantity for the level and the stated marginal quantity for the increment.

Table 7: Frozen arm selection versus the fixed learned-DP column. Predictors and rule choices use fitting requests; all displayed gains use evaluation requests. Gains are percent accepted length at matched budget, with request-clustered 95% intervals. The selected arm and learned DP differ in six of sixteen cells. All windows and both widths are retained; the lower-saturation summary in the main text concerns only a subset.

system	drafts	mode	W	arm the selection picks	its gain	learned DP
27B text	7	greedy	2	learned DP	$+1.31$ $[+1.03, +1.59]$	$+1.31$
27B text	7	greedy	4	predicted-unsat. threshold	$+0.95$ $[+0.71, +1.21]$	$+1.06$
27B text	15	greedy	2	learned DP	$+0.66$ $[+0.27, +1.05]$	$+0.66$
27B text	15	greedy	4	predicted-unsat. threshold	$+0.16$ $[+0.06, +0.27]$	$+0.16$
27B text	7	sampled	2	learned DP	$+1.32$ $[+1.00, +1.63]$	$+1.32$
27B text	7	sampled	4	learned DP	$+1.07$ $[+0.84, +1.29]$	$+1.07$
27B text	15	sampled	2	learned DP	$+0.75$ $[+0.25, +1.21]$	$+0.75$
27B text	15	sampled	4	learned DP	$+0.05$ $[-0.26, +0.35]$	$+0.05$
4B text	7	greedy	2	learned DP	$+0.81$ $[+0.59, +1.04]$	$+0.81$
4B text	7	greedy	4	predicted-unsat. threshold	$+0.37$ $[+0.30, +0.45]$	$+0.44$
4B text	15	greedy	2	learned DP	$+0.45$ $[+0.09, +0.78]$	$+0.45$
4B text	15	greedy	4	predicted-unsat. threshold	$+0.00$ $[-0.00, +0.00]$	-0.14
4B text	7	sampled	2	learned DP	$+0.73$ $[+0.43, +1.00]$	$+0.73$
4B text	7	sampled	4	predicted-unsat. threshold	$+0.26$ $[+0.20, +0.33]$	$+0.23$
4B text	15	sampled	2	learned DP	$+0.46$ $[+0.13, +0.75]$	$+0.46$
4B text	15	sampled	4	predicted-unsat. threshold	$+0.00$ $[+0.00, +0.00]$	-0.41

Table 8: Video signal controls, initial population (16,418 rounds). Confidence is the mean top-1/top-2 margin. Bold marks the focal width event, not a best score. The local-width event has prevalence 9.1%; the sparsity-matched level target has prevalence 11.5%. Intervals cluster on prompt fingerprints. †Learned probes train on two other servers after fingerprint exclusion and evaluate on the full 169,611-round trace. Consequently these columns do not share the hand-signal evaluation population.

	current-round confidence		trailing mean	GBM†	MLP†
predicted quantity	r	AUROC	AUROC	AUROC	AUROC
(control) accepted-length level	$+0.49$ [.47, .50]	0.71 [.70, .72]	0.51 [.50, .52]	—	—
(control) level, sparsity-matched	$+0.46$ [.44, .48]	0.85 [.84, .86]	0.56 [.54, .59]	—	—
width's marginal event	$\mathbf{-0.00}$ [−.02, .01]	$\mathbf{0.48}$ [.46, .49]	0.48 [.46, .49]	0.55 [.54, .57]	0.54
depth's marginal event	$+0.52$ [.49, .53]	0.79 [.78, .80]	0.53 [.51, .55]	0.86	0.83

Interpreting the mechanism. Table 9 reports failure, conditional repair, and a privileged local probe. Conditioning on failure changes the evaluation population. Supplying its position is diagnostic information unavailable to a deployed predictor; a strong privileged score would therefore show remaining structure, not an immediately realizable policy. Conversely, weak scores of these probes do not bound what every possible predictor could achieve. A separate set-valued implementation checks prefix membership rather than reusing only the accepted-length label. The total validation

covers 181,281 rounds; its additional 132 video rounds are not an independent benchmark. The first-increment controls and the positive high-saturation cells are substantive boundaries of the finding, not exceptions removed from the study.

Table 9: Failure–repair diagnostics. Hand columns distinguish failure F_1, repair conditional on failure, and repair using local scores when the failure position is supplied. The local input is privileged. Learned probes use leak-free video training and five-fold request splits on text. The first-increment rows explicitly delimit the deployed-width finding. The set-valued validation uses 132 additional video rounds; it does not define a new independent population.

					AUROC, hand signals			learned	
trace	W	rounds	sat.	$P(E_W)$	F_1	$F_3 \mid F_1$	local at d	F_1	F_3
27B video, 7 drafts	$4 \to 5$	16,418	0.12	0.091	0.87 [0.86, 0.89]	0.54 [0.52, 0.55]	0.53 [0.52, 0.55]	0.89	0.53
27B text, 15 drafts	$4 \to 5$	19,931	0.17	0.104	0.95 [0.94, 0.95]	0.55 [0.54, 0.56]	0.54 [0.53, 0.56]	0.95	0.55
27B text, 7 drafts	$4 \to 5$	20,348	0.53	0.061	0.93 [0.93, 0.94]	0.57 [0.56, 0.59]	0.53 [0.51, 0.54]	0.93	0.57
4B text, 15 drafts	$4 \to 5$	34,496	0.01	0.110	0.95 [0.94, 0.96]	0.51 [0.50, 0.52]	0.54 [0.53, 0.55]	0.95	0.53
4B text, 7 drafts	$4 \to 5$	35,529	0.27	0.087	0.90 [0.89, 0.90]	0.54 [0.53, 0.55]	0.53 [0.52, 0.54]	—	—
27B text, 15 drafts	$1 \to 2$	19,931	0.11	0.329	0.95 [0.95, 0.96]	0.57 [0.56, 0.58]	0.68 [0.67, 0.69]	—	—
4B text, 15 drafts	$1 \to 2$	34,496	0.01	0.337	0.97 [0.96, 0.98]	0.52 [0.52, 0.53]	0.65 [0.64, 0.65]	—	—

D VERIFICATION DEPTH AND INFORMATION SETS

Allocation rule. For each draft position j, a learned survival predictor estimates $\Pr(a \geq j)$. A running minimum enforces non-increasing survival probabilities. The video allocator starts each request at $k_i = 1$ and assigns the remaining $nG - n$ draft slots. The text allocator starts at $k_i = 0$ and assigns all nG draft slots; zero drafts still commit the target's next token. Both allocate by predicted marginal survival while respecting prefix order. For text, the script's variable named G includes the anchor (8), whereas this manuscript's $G = 7$ counts drafts only. The hindsight allocator respects the same lower bound and assigns useful slots with $j \leq a$ before useless slots. Both are evaluated on the same recorded starts and total verification budget. These slots should not be read as a universal measure of tree-kernel runtime.

Previous drafts as information. Table 10 reports the video ladder and its budget sensitivity. The tail comprises positions beyond the preceding acceptance point, discarded when that round ended. Its incremental value over outcome features is $+1.09$ $[0.42, 1.63]$ percentage points; front features add $+0.43$ $[0.04, 0.72]$ points. A separate control run obtains gains of 0.58% for outcome, 1.76% with the true tail, and 0.61% with a tail shuffled conditional on previous accepted length. Shuffled minus true is -1.15 $[-1.61, -0.54]\%$. This control isolates information beyond that acceptance outcome in this population; it does not establish an independent causal mechanism. On the weak EAGLE-3 head with reasoning enabled, a chain experiment at $G = 2$ gives $+0.2\%$ for learned history, $+0.4\%$ with the previous chain, $+1.7\%$ from the first-step distribution, and $+3.6\%$ for the post-draft learner (oracle $+8.8\%$). This is a narrower gain from previous drafts than on the video lattice, not evidence that all autoregressive drafters lack useful history.

Table 10: Video depth ladder. Matched total verification slots $\sum_i k_i = nG$, width-4 tree and seven drafts. Gains are accepted-length changes (%), higher is better; each video request receives at least one draft slot. Intervals at $G = 4$ cluster on prompt fingerprints. Front and tail are aligned to the previous acceptance point; the latter contains discarded draft positions. AUROC refers to acceptance of draft 5. These are per-round measurements, not trajectory or throughput predictions.

signal	side	AUROC	G=3	G=4	G=5	full, G=4
trailing-8 mean	pre	0.528	-0.58	-0.55 $[-1.17, -0.05]$	-0.48	-0.49
stale confidence, untrained	pre	0.634	-1.20	-0.64 $[-1.53, +0.00]$	-0.25	-0.74
learned: accepted-length history	pre	0.575	-0.05	$+0.15$ $[-0.41, +0.78]$	$+0.24$	$+0.08$
+ previous round's outcome	pre	0.607	$+0.67$	$+0.62$ $[+0.03, +1.22]$	$+0.64$	$+0.57$
+ previous lattice, front only	pre	0.637	$+1.02$	$+1.04$ $[+0.28, +1.65]$	$+0.88$	$+1.00$
+ previous lattice, stale tail only	pre	0.670	$+1.71$	$+1.71$ $[+0.71, +2.47]$	$+1.52$	$+1.65$
+ whole previous lattice	pre	0.680	$+1.82$	$+1.98$ $[+1.02, +2.69]$	$+1.68$	$+1.76$
cum. confidence, untrained	post	0.810	$+4.34$	$+4.32$ $[+2.85, +5.45]$	$+3.61$	$+4.41$
learned: current lattice + history	post	0.845	$+6.93$	$+6.33$ $[+4.91, +7.40]$	$+4.56$	$+6.16$
hindsight oracle	—	—	$+16.25$	$+13.24$	$+7.44$	$+12.77$

Target information and heterogeneity. Table 11 reports pooled information sets; Table 12 retains all five benchmarks. The pre-draft lattice and target rungs each add to history/outcome and are not nested in one another. The post-draft input is current-lattice features plus the four history features; it does not include the separate previous-outcome column or target features. A pooled row reallocates across a mixed-benchmark pool; it is not an average of the five within-benchmark rows. On MBPP, GSM8K, and MATH-500, target-state gain divided by post-draft gain is 0.46 $[0.38, 0.52]$ on 27B and 0.50 $[0.45, 0.53]$ on 4B, versus 0.28 and 0.22 for the previous lattice. These ratios concern this selected, comparatively homogeneous three-benchmark group. HumanEval and MT-Bench show different values and are included in the main figure. The pooled mixed-workload ratio is not an estimate for a typical single workload.

Requiring 15 subsequent committed tokens excludes roughly 5–9% of rounds on HumanEval, MBPP, and GSM8K. Keeping end-of-output rounds raises the three-benchmark target/post ratios to 0.51 $[0.44, 0.56]$ and 0.54 $[0.50, 0.58]$. Thus the ordering persists, but the precise fraction depends on endpoint handling. Two-round-old target features retain useful gains (Table 11), motivating delayed execution without establishing its live cost.

Table 11: Text information-set comparison on the re-collected traces. Accepted-length gain (%) over uniform verification depth at a mean of 7 of 15 drafts; random groups of eight request-rounds, allowing zero draft slots per round, and request-clustered intervals. AUROC predicts $a \geq 8$. Target state means history plus previous target logits and a hidden-state projection. Pre-draft inputs add separately to history/outcome. Post-draft also uses history, but not the separate outcome column. The bottom block uses two-round-old features. Rows require a complete 15-token continuation.

	27B hybrid target			4B target		
signal	AUROC	greedy gain	sampled gain	AUROC	greedy gain	sampled gain
history + outcome	0.79	+8.8 [+7.0, +10.9]	+9.7	0.75	+3.0 [+2.4, +3.6]	+2.8
+ previous lattice	0.83	+11.1 [+9.3, +13.1]	+12.3	0.79	+3.8 [+3.2, +4.3]	+3.9
+ target state	0.86	+13.2 [+11.3, +15.3]	+14.7	0.85	+5.5 [+4.9, +6.1]	+5.4
+ both	0.87	+13.4 [+11.5, +15.5]	+14.9	0.85	+5.5 [+4.9, +6.1]	+5.5
post-draft learned	0.94	+20.1 [+18.2, +22.2]	+21.5	0.93	+8.3 [+7.6, +8.9]	+8.0
hindsight oracle	—	+27.7 [+26.3, +29.3]	+28.6	—	+10.5 [+9.8, +11.2]	+10.0
two rounds old: history	0.75	+7.2 [+5.4, +9.2]	+8.0	0.71	+2.2 [+1.6, +2.6]	+2.0
two rounds old: + target state	0.81	+10.2 [+8.3, +12.4]	+11.4	0.79	+4.0 [+3.5, +4.5]	+4.0

Table 12: All five benchmarks for the re-collected information ladder. H: history and outcome; L: H plus previous lattice; T: H plus target features; L+T: both; post: current lattice plus history. The pooled row reallocates across benchmarks rather than averaging their rows. Gains are percent accepted length at a fixed verification budget. The ratio is (L+T)/post, not T/post. Only rounds with at least 15 subsequent committed tokens are included; end-of-output sensitivity is discussed in the text.

	27B hybrid target						4B full-attention target					
batches from	H	L	T	L+T	post	ratio	H	L	T	L+T	post	ratio
GSM8K	+2.9	+5.1	+8.1	+8.4	+15.4	54%	+1.2	+2.3	+4.1	+4.1	+8.4	49%
MATH-500	+2.3	+4.1	+7.1	+7.2	+15.4	47%	+1.8	+2.8	+5.6	+5.8	+11.3	51%
MBPP	+1.1	+3.3	+5.8	+5.9	+14.2	42%	−0.1	+0.5	+1.8	+1.9	+3.6	51%
HumanEval	+8.2	+9.8	+11.7	+11.7	+15.7	75%	+2.9	+4.2	+6.3	+6.5	+9.4	70%
MT-Bench	+6.4	+9.5	+9.8	+10.3	+13.6	76%	+2.0	+2.6	+3.1	+3.1	+4.1	74%
all five, pooled	+8.8	+11.1	+13.2	+13.4	+20.1	66%	+3.0	+3.8	+5.5	+5.5	+8.3	67%

E NODE-BUDGET CONTROLS AND EXPLORATORY ASSOCIATION

EAGLE-3 coverage. The weak public head is `AngelSlim/Qwen3-4B_eagle3`; the stronger authors' head is `yuhuili/EAGLE3-LLaMA3.1-Instruct-8B`. The weak-head collection uses 1,671 requests: 500 GSM8K, 500 MATH, 427 MBPP, 164 HumanEval, and 80 MT-Bench, up to 512 new tokens at concurrency eight. It contains 161,774 rounds in 21,088 batches, of which 21,052 are multi-request batches. Replay uses 160,103 consecutive pairs, dropping approximately 0.8% that cannot be aligned. Its candidate pool has 392 nodes, from top-eight expansion over seven steps, with 31 nodes verified in the default trace.

The stronger head is a necessary control against attributing weak-head behavior to the method family. An 80-prompt MT-Bench gate obtains mean accepted length 5.922 in the fork and 5.396 in the authors' code, but uses different tree depths (eight versus five). This establishes a functional head after tokenizer correction, not matched superiority of the fork. The main node-budget replay uses its stated 31-node setup, not the gate's distinct configuration.

Action-set-matched oracles. Table 13 distinguishes integer-budget and smaller action-menu oracles. On the weak head, calibrated allocation gains of 1.74, 0.96, and 0.66% recover 14.6, 9.8, and 8.1% of the menu oracle, versus 12.6, 8.2, and 6.7% of the integer-budget oracle. The menu oracle itself yields 11.88, 9.86, and 8.08%, versus 13.83, 11.73, and 9.80% for unrestricted integer budgets. Frozen-half selection gives 1.75 [1.63,1.86], 0.92 [0.84,1.01], and 0.69 [0.62,0.75]%, addressing selection optimism in the five-fold policy comparison.

Table 13: Weak EAGLE-3 head on Qwen3-4B: node-budget prediction and allocation. Accepted-length gains (%) compare with uniform per-request budget at the same total; intervals cluster on requests. Both unrestricted integer-budget and action-menu hindsight oracles are shown. The learned-policy row is the original menu regressor; calibrated policies discussed in the text are separate runs.

	$n_0 = 8$	$n_0 = 16$	$n_0 = 31$ (served)
AUROC level: exp. accepted length	0.703	0.685	0.673
AUROC margin: exp. accepted length	0.512	0.506	0.533
AUROC margin: chain confidence	0.515	0.505	0.530
AUROC margin: score of node $n_0{+}1$	0.545	0.538	0.553
AUROC margin: score mass, next n_0	0.538	0.538	0.555
AUROC margin: trailing mean	0.510	0.507	0.513
AUROC margin: learned	0.589	0.569	0.566
prevalence of the marginal event	22.9%	20.8%	19.6%
uniform τ	2.797	3.074	3.307
oracle, integer budgets	$+13.83$ $[+13.68, +13.99]$	$+11.73$ $[+11.60, +11.86]$	$+9.80$ $[+9.70, +9.90]$
oracle, menu budgets	$+11.88$ $[+11.73, +12.03]$	$+9.86$ $[+9.74, +9.97]$	$+8.08$ $[+8.00, +8.18]$
batch-global rerank	-0.19 $[-0.30, -0.09]$	-0.14 $[-0.21, -0.07]$	-0.09 $[-0.15, -0.03]$
learned policy (menu)	$+0.41$ $[+0.33, +0.48]$	-0.21 $[-0.27, -0.15]$	-0.36 $[-0.41, -0.31]$
$\propto$ score mass, next n_0	-0.34 $[-0.44, -0.24]$	-0.43 $[-0.52, -0.35]$	-0.37 $[-0.45, -0.30]$
$\propto$ exp. accepted length	-0.61 $[-0.72, -0.51]$	-0.77 $[-0.86, -0.68]$	-0.43 $[-0.51, -0.36]$
$\propto$ its inverse	-1.12 $[-1.21, -1.04]$	-1.11 $[-1.19, -1.03]$	-0.93 $[-1.00, -0.86]$
$\propto 1-$ chain confidence	-0.41 $[-0.46, -0.35]$	-0.54 $[-0.60, -0.48]$	-0.45 $[-0.50, -0.40]$
$\propto$ chain confidence	-12.04 $[-12.32, -11.78]$	-13.62 $[-13.92, -13.34]$	-14.36 $[-14.66, -14.08]$

Association is not a law. Table 14 contains all 24 cells. The four starved-budget cells all use eight nodes and a 15-draft window, and include two Muse splits. Their 5.52–14.26% gains are positive results, but budget changes jointly affect event definition, oracle headroom, and predictability. The overall correlation between margin AUROC and recovered integer-oracle fraction is 0.897; within-cluster and between-cluster correlations are 0.788 and 0.944. A regression over the 20 cover-budget cells gives slope 68.31 percentage points per AUROC unit. There are only five drafter/target clusters: CR1 inference gives $t = 9.72$ (t_4 two-sided $p \approx 0.0006$), whereas enumerated wild-cluster signs give $p = 0.0625$. We retain both rather than infer a universal threshold from the smaller p-value. Fourteen cover-budget cells use synthetic batches.

The gap between menu and integer-budget oracles ranges from 1.26 to 8.32 percentage points in the relevant comparison. Consequently, a recovered fraction using the integer oracle can understate what a predictor achieves within its available actions. Recovered fractions should therefore be reported with an action-set-matched oracle; the cross-cell association does not identify a causal effect of AUROC on gain.

Table 14: Exploratory node-budget comparison: all 24 cells. AUROC predicts whether doubling a request's node budget increases acceptance; allocator and oracle columns are percent accepted-length gain over uniform. Arms use five-fold request evaluation, and lattice traces use synthetic batches of eight. Share is relative to the unrestricted integer-budget oracle and is descriptive, not a bound on the tested signal. Some allocators use a smaller menu. Recalibration is the gain over raw reranking. Calibration and held-out Muse splits are not independent systems.

drafter, target	P	split	n_0	AUROC	allocator	oracle	share (%)	recal.
budget covers the draft window ($n_0 \geq P$): every configuration deployed here								
EAGLE-3 head, Llama-3.1-8B	7	—	8	0.708	+2.54	+14.43	17.6	+0.01
EAGLE-3 head, Llama-3.1-8B	7	—	16	0.705	+1.57	+10.40	15.1	+0.02
EAGLE-3 head, Llama-3.1-8B	7	—	31	0.722	+1.16	+6.51	17.8	+0.01
EAGLE-3 head, Qwen3-4B	7	—	8	0.589	+1.74	+13.83	12.6	+1.93
EAGLE-3 head, Qwen3-4B	7	—	16	0.569	+0.96	+11.73	8.2	+1.10
EAGLE-3 head, Qwen3-4B	7	—	31	0.566	+0.66	+9.80	6.7	+0.74
lattice, 27B text	7	—	8	0.745	+1.88	+9.41	20.0	+0.13
lattice, 27B text	7	—	16	0.735	+1.54	+6.99	22.1	+0.20
lattice, 27B text	7	—	31	0.741	+1.19	+5.13	23.1	+0.18
lattice, 27B text	15	—	16	0.635	+2.24	+11.80	19.0	+0.37
lattice, 27B text	15	—	31	0.612	+1.07	+8.50	12.5	+0.58
lattice, 4B text	7	—	8	0.651	+1.89	+12.99	14.6	+0.13
lattice, 4B text	7	—	16	0.633	+1.06	+10.07	10.5	+0.19
lattice, 4B text	7	—	31	0.644	+0.83	+7.96	10.4	+0.13
lattice, 4B text	15	—	16	0.579	+1.21	+12.28	9.8	+0.14
lattice, 4B text	15	—	31	0.549	+0.60	+9.97	6.0	+0.15
lattice, Muse-30B	15	calib	16	0.587	+1.81	+13.97	12.9	+0.46
lattice, Muse-30B	15	calib	31	0.558	+0.87	+10.83	8.0	+0.26
lattice, Muse-30B	15	held	16	0.585	+1.62	+13.82	11.7	+0.06
lattice, Muse-30B	15	held	31	0.567	+0.74	+10.70	6.9	+0.12
budget below the window ($n_0 < P$): a starved tree, never deployed here								
lattice, 27B text	15	—	8	0.813	+14.26	+26.42	54.0	+0.07
lattice, 4B text	15	—	8	0.676	+5.52	+19.19	28.8	+0.06
lattice, Muse-30B	15	calib	8	0.679	+8.07	+24.20	33.3	+0.51
lattice, Muse-30B	15	held	8	0.675	+8.76	+24.49	35.8	+0.39

F LIVE PROTOCOLS AND BASELINE ACCOUNTING

Controlled horizon experiment. The calibration set contains 120 requests and the held-out set 400 disjoint requests, split between Dolly chat and GSM8K-train. Native graph execution, equal admission, and rotated GPUs are used. Histories and target-state predictors can act without a fresh host read; post-draft confidence uses one. Constant-action controls retain adaptive execution machinery. The control at the best horizon uses $\gamma = 12$ at concurrency one and 8 at concurrency eight. The observational control uses $\gamma = 16$ at both loads. Table 15 reports the measured constant-action curve and prediction on identical observational rounds. Predicting accepted length on those rounds is not estimating the effect of changing horizon.

The earlier target-head experiment and the controlled experiment use different trained heads. The earlier full head includes logit-derived features; a folded head is refit without them to avoid an additional synchronization. Neither is the offline depth gradient-boosted predictor. Thresholds select among horizons 8, 12, and 16. The cost-aware rule calibrates the target signal to expected accepted length at each horizon and divides by measured round time; it falls back to the best constant unless the predicted gain exceeds 1%. Per-request results, policy parameters, and executed horizon records are retained with the controlled experiment, so that action distributions can be checked rather than inferred from arm names.

Table 15: Controlled horizon experiment: held-out cost curve and signal prediction on identical observational-arm rounds. The best constant depends on concurrency. Signal R^2 predicts accepted length at fixed horizon 16, not the gain from changing horizon. Upper rows report system tokens/s, not the request-rate estimator R of the policy comparisons. Bold marks the best constant and, separately, the highest signal R^2.

load	quantity	γ=8	γ=12	γ=16
c=1	round time (ms) / τ / tok/s	15.14 / 4.211 / 278.2	16.09 / 4.538 / **282.1**	16.94 / 4.584 / 270.5
c=8	round time (ms) / τ / tok/s	27.80 / 4.207 / **1210.3**	31.25 / 4.554 / 1165.9	33.73 / 4.611 / 1093.8
	signal (R^2, fixed action)	post-draft conf.	target state	history
c=1		**0.289**	0.257	0.221
c=8		**0.305**	0.270	0.231

Observation decomposition. The later fixed-$\gamma = 16$ experiment keeps mean accepted length at 4.5836 across arms. Machinery alone changes throughput by $+0.04$ $[-0.57, 0.65]\%$, the added host read by -4.23 $[-4.70, -3.75]\%$, and head/logging beyond that by -1.26 $[-1.56, -0.95]\%$. The original fixed-action concurrency-eight control has a small residual acceptance difference; its throughput change is therefore not a pure latency charge. Cross-wave drift further prevents using the earlier best-horizon control and later horizon-16 control as a clean amortization curve.

Static comparators. For request sets with separable cost and unchanged token totals, let $C_w(h)$ be aggregate latency for workload w under constant horizon h. The ideal workload lookup has cost $\sum_w \min_h C_w(h)$, at most $\min_h \sum_w C_w(h)$ for a single global choice. This elementary inequality explains why the global baseline can be weak. It does not give an observed speedup, include label/engine overhead, or justify recombining batched requests. Adding a new horizon can improve either optimum, so it need not increase the lookup's relative advantage. For example, costs $(1, 3)$ and $(3, 1)$ favor workload-specific choice, but a new horizon costing $(0.9, 0.9)$ eliminates that advantage.

Table 16 preserves the seven-stream retrospective comparison. Its hindsight workload choices differ from the main text's calibration-frozen lookup. The later served lookup also uses two routed engines and a new measurement wave; its same-wave target-policy comparison is the relevant contrast. A free workload label is an assumption, and resource equivalence is not established merely by charging the measured controller overhead.

Ragged verification. Table 17 reports the full-attention 4B prototype. Verification block size $b = k + 1$ includes the anchor. At concurrency 16, history, target state, and post-draft confidence change throughput by -16.1, -29.7, and -2.3%. Execution is eager, and the calibration projection and executed comparison use different best uniform blocks. The prototype therefore does not isolate

Table 16: History horizon rule against three baselines on seven streams, concurrency one. The workload constants here are hindsight choices, reselected within each bootstrap; calibration-frozen and served choices are evaluated separately in the main text. ^{r}Launches were pooled in reconstructed scheduler records; ^{m}Muse, other rows 27B. Best-global and ‡ entries have two-level intervals; remaining entries have prompt-level intervals. Gains use request-rate R. Default-relative gains are not AR-relative speedups.

stream (n requests)	vs. default γ=8	vs. best single constant	vs. constant per workload
five benchmarks, in fit (160)^{r}	$+10.0$ $[+7.8, +12.5]$	$+2.4$ $[+0.5, +3.9]$	$+0.7$ $[-1.0, +1.8]$
four benchmarks, held out (128)^{r}	$+17.2$ $[+14.0, +20.4]$	$+1.1$ $[-0.7, +2.8]$	$+0.9$ $[-1.0, +2.3]$
four workloads incl. chat (400)	$+7.6$ $[+6.4, +8.9]$	$+0.76$ $[-0.01, +1.53]$	$+0.18^{\ddagger}$ $[-0.48, +0.83]$
predictable: chat + GSM8K (1,200)	$+3.3$ $[+2.8, +3.8]$	$+1.61$ $[+1.03, +2.18]$	-0.4 $[-0.7, +0.0]$
unpredictable: code + maths (914)	$+13.0$ $[+12.3, +13.7]$	-0.13 $[-0.56, +0.31]$	-0.1 $[-0.6, +0.2]$
Dolly chat + GSM8K train (1,200)	$+2.8$ $[+2.3, +3.3]$	$+1.14$ $[+0.53, +1.76]$	$-0.80^{\ddagger}$ $[-1.29, -0.31]$
five workloads (160)^{m}	$+10.8$ $[+9.0, +12.7]$	-3.19 $[-4.28, -2.10]$	-3.3 $[-4.5, -2.4]$

the online value of the offline depth allocator, nor establish that graph-compatible ragged execution is unavailable in other systems.

Table 17: Live verification-depth prototype on the full-attention 4B system, eager execution, equal admission, 400 held-out requests and three launches. Gains in request-rate R compare with the best measured uniform block at that load; tok/s is system throughput, a different aggregation. Bold identifies the chosen uniform baseline, not the best overall arm. The calibration projection uses a different best uniform block and is not an executed result. ^{p}Intervals omit the launch component; including it makes both uniform-ordering comparisons inconclusive. $b = k + 1$ counts verified tokens including the anchor.

		c=8		c=16	
arm	note	tok/s	gain	tok/s	gain
uniform	b=16	566.6	-1.2 $[-1.6, -0.7]^{p}$	**1216.1**	—
uniform	b=12	**572.9**	—	1196.4	-1.9 $[-2.7, -1.1]^{p}$
ragged, target's state	proj. -1.6 / -7.9	476.9	-17.0 $[-19.3, -14.8]$	860.1	-29.7 $[-31.2, -28.1]$
ragged, history EMA	proj. -2.0 / -9.0	545.1	-5.1 $[-5.8, -4.4]$	1021.9	-16.1 $[-17.8, -14.4]$
ragged, post-draft conf.	one sync	575.3	$+0.4$ $[-0.6, +1.4]$	1187.1	-2.3 $[-3.4, -1.3]$

G Uncertainty and Multiple Comparisons

Offline intervals resample requests or video prompt fingerprints after predictions and synthetic group assignments have been fixed; they do not include retraining or regrouping variation. The live comparison metric R is aggregate tokens divided by aggregate per-request latency, under the stated load. It is neither a mean of request-level speedup ratios nor, with overlapping requests, wall-clock system throughput. Tables 15 and 17 separately retain system tokens/s from the benchmark; differences between their token-rate ratios and reported R gains need not be rounding. The primary two-level calculation adds prompt and launch components, $\widehat{\mathrm{SE}}^2 = \widehat{\mathrm{SE}}_{\mathrm{prompt}}^2 + s_{\mathrm{launch}}^2/L$, using the recorded launches and paired prompts where available. Normal 95% intervals are marginal intervals; Holm correction acts on the comparison family, not separately on each displayed table.

The original 43-test family is retained: 33 earlier live contrasts, eight controlled-horizon contrasts, and two served-static contrasts. Table 18 retains the 33 earlier rows with adjustment over all 43. Under the primary convention, 27 comparisons survive, six with positive gains. A hierarchical-bootstrap hybrid sensitivity retains 26. Applying Satterthwaite small-sample correction to all ten later two-launch tests retains 25, with the same six positives. For the controlled target-state policy, the adjusted p-value changes from 0.0019 to 0.0046; for the served lookup it is 0.0010 under that sensitivity. The unadjusted history interval above zero does not imply family-wise significance (adjusted $p = 0.14$).

Two launches weakly estimate between-launch variance; many requests do not resolve that limitation. A single-launch interval describes prompt variation only. These sensitivities improve transparency but do not replace additional independent launches. The dose-response analysis uses only five system clusters and is interpreted separately, as described in Appendix E.

Table 18: The original 33 live comparisons, adjusted over the retained family of 43. The other eight are the controlled horizon policies and the remaining two are the served workload-static comparisons. Holm uses two-level variance-added intervals; Holm_{h} uses a hierarchical-bootstrap sensitivity for these rows and normal two-level values for the ten later tests. ^{s}One measured launch; ^{t}two-launch Muse estimate, interpreted descriptively; ^{r}passes pooled before record reconstruction. The survival column belongs to the primary convention, not to both sensitivities. Gains use R, summed tokens over summed per-request latency.

comparison	gain	interval	raw p	Holm	Holm_{h}	surv.
27B, four workloads, fresh (400)						
history rule vs best single constant	$+0.76$	$[-0.01, +1.53]$	0.053	0.59	0.61	no
history rule vs constant per workload	$+0.18$	$[-0.48, +0.83]$	0.594	1	1	no
history rule vs best single constant, $c{=}8^{s}$	-2.07	$[-3.61, -1.17]$	$2{\cdot}10^{-4}$	0.0044	0.0042	**yes**
27B, predictable stream (1,200)						
history rule vs best single constant	$+1.61$	$[+1.03, +2.18]$	$<4{\cdot}10^{-5}$	0.0015	0.0015	**yes**
same, $c{=}8^{s}$	-3.01	$[-3.59, -2.41]$	$<4{\cdot}10^{-5}$	0.0015	0.0015	**yes**
27B, unpredictable stream (914)						
history rule vs best single constant	-0.13	$[-0.56, +0.31]$	0.563	1	1	no
difference of the two gains	$+1.73$	$[+0.99, +2.48]$	$<4{\cdot}10^{-5}$	0.0015	0.0015	**yes**
27B, Dolly chat + GSM8K train (1,200)						
history rule vs best single constant	$+1.14$	$[+0.53, +1.76]$	$3{\cdot}10^{-4}$	0.0047	0.0015	**yes**
history rule vs constant per workload	-0.80	$[-1.29, -0.31]$	0.001	0.024	0.012	**yes**
per-workload constants vs best single	$+1.96$	$[+1.50, +2.43]$	$<4{\cdot}10^{-5}$	0.0015	0.0015	**yes**
27B, five benchmarks, prompts in the fit (160)^{r}						
history rule vs best single constant	$+2.43$	$[+0.53, +3.88]$	0.014	0.20	0.18	no
stack's acceptance-EMA controller vs same	-4.26	$[-6.71, -2.22]$	$2{\cdot}10^{-4}$	0.0044	0.0042	**yes**
27B, four benchmarks, held out (128)^{r}						
history rule vs best single constant	$+1.06$	$[-0.66, +2.79]$	0.229	1	1	no
stack's acceptance-EMA controller vs same	-8.13	$[-10.13, -6.07]$	$2{\cdot}10^{-4}$	0.0044	0.0042	**yes**
27B, five workloads, fresh (192): target state						
full head vs best single constant	-1.34	$[-2.56, -0.12]$	0.032	0.41	0.14	no
full head − history rule	-2.35	$[-3.60, -1.10]$	$2{\cdot}10^{-4}$	0.0044	0.0015	**yes**
history rule vs best single constant (block 1)	$+1.04$	$[-0.28, +2.36]$	0.122	1	1	no
folded head vs best single constant	$+0.08$	$[-1.16, +1.31]$	0.903	1	1	no
folded head − history rule	-0.81	$[-1.93, +0.30]$	0.151	1	1	no
history rule vs best single constant (block 2)	$+0.90$	$[-0.40, +2.20]$	0.176	1	1	no
4B, four workloads (400)						
history rule vs best single constant	-2.97	$[-4.28, -1.65]$	$<4{\cdot}10^{-5}$	0.0015	0.0015	**yes**
history rule vs constant per workload	-2.97	$[-4.28, -1.65]$	$<4{\cdot}10^{-5}$	0.0015	0.0015	**yes**
history rule vs best single constant, $c{=}8^{s}$	-0.56	$[-1.65, +0.55]$	0.317	1	1	no
Muse-Glimmer-30B, five workloads (160)						
history rule vs best single constant	-3.19	$[-4.28, -2.10]$	$<4{\cdot}10^{-5}$	0.0015	0.0015	**yes**
same, $c{=}8^{t}$	-2.71	$[-5.31, -0.10]$	0.042	0.50	0.17	no
4B ragged verification, $c{=}8$ (400)						
history EMA vs best uniform count	-5.07	$[-5.78, -4.36]$	$<4{\cdot}10^{-5}$	0.0015	0.0015	**yes**
target-state head vs best uniform count	-17.04	$[-19.33, -14.75]$	$<4{\cdot}10^{-5}$	0.0015	0.0015	**yes**
target-state head − history EMA	-12.61	$[-14.92, -10.30]$	$<4{\cdot}10^{-5}$	0.0015	0.0015	**yes**
post-draft confidence vs best uniform count	$+0.39$	$[-0.64, +1.42]$	0.462	1	1	no
4B ragged verification, $c{=}16$ (400)						
history EMA vs best uniform count	-16.13	$[-17.82, -14.43]$	$<4{\cdot}10^{-5}$	0.0015	0.0015	**yes**
target-state head vs best uniform count	-29.67	$[-31.23, -28.11]$	$<4{\cdot}10^{-5}$	0.0015	0.0015	**yes**
target-state head − history EMA	-16.15	$[-17.76, -14.53]$	$<4{\cdot}10^{-5}$	0.0015	0.0015	**yes**
post-draft confidence vs best uniform count	-2.32	$[-3.36, -1.27]$	$<4{\cdot}10^{-5}$	0.0015	0.72	**yes**

H Related Work and Comparison Scope

Exact speculative sampling preserves the target distribution by construction (Leviathan et al., 2023; Chen et al., 2023). EAGLE-2 dynamically shapes candidate trees and EAGLE-3 improves the draft model (Li et al., 2024; 2025); these advances do not imply that every further cross-request reallocation is profitable. Opt-Tree addresses tree construction (Wang et al., 2025). Our within-request best-first control is essential for separating that problem from the cross-request budget decision.

TETRIS allocates draft tokens across requests (Wu et al., 2025), while LibraSpec explicitly connects marginal acceptance and verification cost (Lin et al., 2026). These are direct precedents for the allocation objective. Our failure–repair analysis and controlled comparisons study the empirical information supporting it; they do not claim to introduce marginal-gain optimization. Bole also addresses hybrid-model verification execution (Wang et al., 2026), so limitations of our hybrid-model serving path are not architecture-wide impossibility results.

AdaEAGLE predicts draft length (Zhang et al., 2024); BlockPilot chooses a request's block size once after prefill (Zhang et al., 2026). Their predictors motivate the target-state and stronger-static controls here. Published speedups depend on the sampling configuration and comparator, and cannot be ranked against our percentages without matched models and hardware.

DSpark uses delayed capacity estimates with current candidate ranking (Cheng et al., 2026). D-cut packs variable-depth verification into graph buckets (Liu et al., 2026b); AdaFlash combines adaptive lengths and packing with on-policy drafter training (Qian et al., 2026). These designs make runtime scheduling part of the method. Our host-read control measures one implementation path, not an unavoidable cost of their information.

Two recent studies are especially close in motivation. Luo et al. (2026) compare confidence and learned survival signals under one lossy verifier-skipping policy and show that better positionwise prediction need not improve scheduling. We study target-verified candidate allocation, including pre-draft information and local width repair. CURE uses confidence margins to locate fragile positions and generate bounded repair branches (Liu et al., 2026a). That changes the candidate construction, whereas our local-width comparison holds the lattice fixed. Neither setting is invalidated by the other's results. A matched implementation comparison with these policies remains outside our experiments.

I Prospective Tests, Deviations, and Invalidated Runs

The experiments combine exploratory analysis, frozen held-out tests, and locally recorded prospective predictions. Local protocol files are not independent public preregistrations. The saturation cutoff of 0.4 was chosen after inspection. The initial seven-draft width null failed on the saturated 27B trace, and the prospective stronger-head prediction (AUROC at most 0.65 and gain at most 2%) failed. The positive results are retained in the final account.

A three-system prospective sign test gives live gains of $+0.76$ $[-0.01, 1.53]\%$ on 27B, -2.97 $[-4.28, -1.65]\%$ on 4B, and -3.19 $[-4.28, -2.10]\%$ on Muse under the final two-level calculation. The original prompt-only 27B interval barely excluded zero. This supports a distinction between meeting a recorded prompt-level criterion and obtaining robust launch-aware evidence; it does not establish calibrated magnitude prediction.

A later prospective test requires the policy's margin over a frozen workload lookup to change as specified between new streams. Its launches failed twice with filesystem errors and the test is incomplete. A reconstructed lookup advantage of approximately 2.01% on one stream and zero by construction on another does not test the required adaptive-policy margin. No success is claimed from those static calculations.

Invalidated development runs are excluded: an early live run used scripts modified while arms were running and did not execute the intended action distributions; executed-horizon checks revealed the problem. An initial strong-head gate duplicated the beginning-of-sequence token and was rerun after token-ID correction. An early greedy oracle was replaced with exact discrete optimization because actual curves need not be concave. Later control artifacts include hashes and smoke checks, but those records should not be described as externally certified registrations.

J CLAIM-TO-ARTIFACT INDEX

Paths below are relative to the supplementary archive's `v12/` directory. They identify existing evidence used in this manuscript rather than experiments newly performed for its presentation. The accompanying editorial source includes the source-PDF hash, retained table cells, figure values, and an evidence map. The small supplementary archive does not contain the multi-gigabyte raw trace collections. Historical analysis scripts require configured data/model paths and the serving environment. The separate Figure 2 evidence package contains numeric request IDs, accepted lengths, and saved survival predictions, with a standalone recomputation script; it verifies allocation summaries without retraining models or replaying GPU execution.

Evidence	Artifact location
Frozen width policies and held-out gains	`c1_width/`, including `PROTOCOL_frozen.md`, `video_frozen.json`, and `frozen_*.json`
Depth, previous-lattice information, consistency	`c2_depth/`, `c3_consistency/`, `c4_consistency/`
Executed-width checks	`e1_samepfx/`
Prospective signs and target live heads	`e2_prospective/`, `e3_tgtlive/`
Workload baselines and additional streams	`e6_perworkload/`, `e7_summ_longctx/`
Controlled horizon policies	`rev7_p0/p1b_controller/p1b_results.json`
Served frozen workload lookup	`rev7_p1/pwc/pwc_live.json`, with frozen launch scripts
Cluster sensitivity for node budgets	`rev7_p1/dose/dose_cluster.py`