TRAINING RESHAPES WIDTH-DEPENDENT ACCEPTANCE IN BLOCK-DIFFUSION SPECULATIVE DECODING

Anonymous authors
Paper under double-blind review

ABSTRACT

Inference-time width selection balances a drafter's acceptance against execution cost. We study how training changes that balance. In block-diffusion drafting, a narrower forward can lower acceptance even when the verified prefix is unchanged. A three-arm experiment separates training forward width from the positions included in the loss. With identical prefix supervision, narrow-layout training improves acceptance at narrow evaluation widths and lowers it at a wider width in all three paired seeds. On 900 prompts from six sources, the gain extends to width 8, the configuration selected for serving. This finding motivates cost-driven adaptation: identify promising training layouts from measured costs, adapt with standard cross-entropy, then select each head's deployment width independently. On MI308X, both supervision-matched heads select width 8 from the same candidates. Narrow-layout training improves throughput by 1.58% on average, with positive gains in all three seeds and sessions and 17 of 18 paired comparisons during nine-minute runs. Public-checkpoint and cross-platform tests show how the benefit varies across models and workloads. Training thus changes the acceptance–cost trade-off available to a deployment policy.

1 INTRODUCTION

Speculative decoding accelerates generation by verifying several draft tokens in one target-model pass (Leviathan et al., 2023; Chen et al., 2023). Block-diffusion drafters predict these candidates together in a non-causal forward pass (Chen et al., 2026). Smaller blocks reduce the work per round, but can also change predictions at the first few positions, because positions within a block can attend to one another. A shorter accepted prefix may therefore reflect both fewer candidates and poorer predictions among those we keep. Width selection balances these predictions against execution cost for a fixed drafter. We ask whether training can improve the acceptance profile on which that choice depends.

The same candidate positions are predicted better in a wider forward. With a frozen head and verification capacity 8, a width-8 forward yields 3.6146 committed tokens per forward, while a width-17 forward yields 3.7337. The narrower layout loses 3.19% even though both runs verify the same prefix positions. The extra positions in the wider forward are never verified, yet their presence changes the acceptance of the retained prefix.

Training shifts acceptance toward narrower layouts. To separate forward width from the loss positions, we train width-10 and width-16 heads with the same nine supervised positions, sequence order, block starts, and update budget. Across three paired seeds and 900 prompts, narrower-layout training improves acceptance at widths 8 and 10 and lowers it at width 17, with verification capacity fixed at 8 throughout. The two wide-trained controls have nearly overlapping profiles; the narrow-trained head retains higher acceptance at narrow evaluation widths. The acceptance profile changes even though the supervised positions stay the same.

Hardware costs help us choose where to adapt. The width with better acceptance is not always the width that runs faster. On MI308X at batch 16, increasing query width from 9 to 10 reduces drafting time from 8.642 to 7.213 ms. We use measured costs to find useful execution widths, train the drafter at those layouts, and select a serving configuration for the adapted head. Both supervision-matched

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.

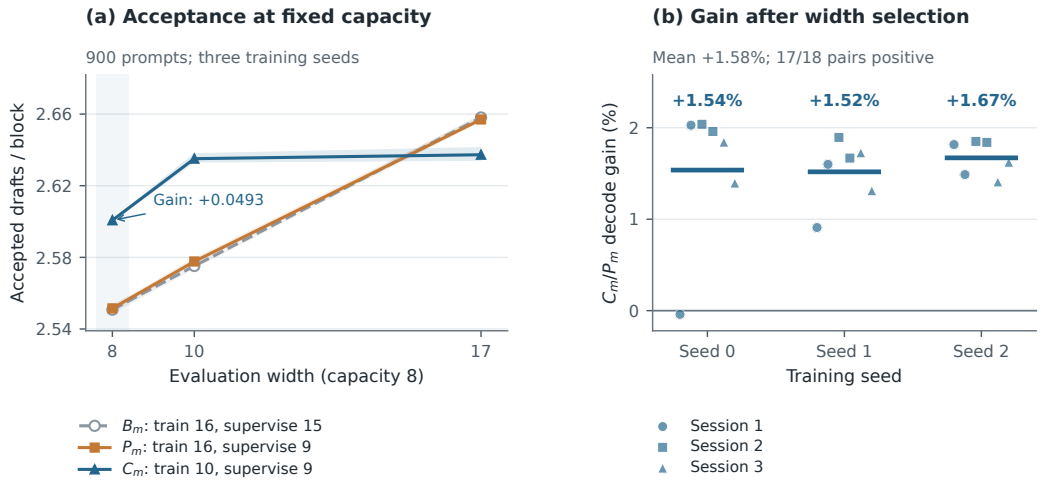

Figure 1: **Training reshapes acceptance and improves serving.** (a) Three training arms on 900 prompts at capacity 8. P_m/C_m share nine-position supervision; B_m supervises all positions. Lines join means across three seeds; bands span their minimum and maximum. The C_m advantage extends to the selected width 8. (b) Online gains for the same P_m/C_m heads after independent width selection. Points are paired runs, shapes identify sessions, and horizontal bars show seed means.

heads independently select width 8 for serving. At this width, the narrow-trained head improves offline acceptance in all three seeds and six sources, and online throughput by 1.58% across three seeds in nine-minute runs. Training width shapes the acceptance profile; deployment width selects an operating point on the resulting profile.

DFlash studies training–inference block-size mismatch (Chen et al., 2026), and BlockPilot learns to select inference widths (Zhang et al., 2026). We separate forward layout from two quantities that usually change with width: the number of verified candidates and the positions supervised during training. Holding both fixed reveals a training-layout effect on retained-prefix acceptance, which we then trace through to serving after each head selects its deployment width. Figure 1 summarizes the progression from the controlled layout effect to serving gains; Sections 4–7 present the evidence.

Our contributions are:

1. **Layout affects the retained prefix.** At fixed verification capacity, changing forward width changes acceptance at the same candidate positions.
2. **Forward layout changes learning under matched supervision.** With supervised positions and verification capacity fixed, narrow-layout training improves acceptance at widths 8 and 10 and lowers it at width 17 in all three seeds on 900 prompts. Public checkpoints provide additional matched-layout gains on two model sizes.
3. **The changed profile improves serving after width selection.** The same supervision-matched heads gain throughput in all three training seeds and all three sessions after independent, equal-opportunity width selection. Further experiments examine joint deployment and workload dependence.

2 Training an Acceptance Profile for Deployment

We use width at two stages. Measured execution costs guide the choice of a training layout, which changes the drafter's acceptance profile. We then select a serving width for the trained head. The two widths serve different purposes and are chosen separately.

2.1 Width Is Both a Model Layout and an Execution Shape

We distinguish training width w_{train}, draft-forward width w, and verification capacity k. A block contains one anchor and $w-1$ proposal positions; capacity k permits at most $k-1$ draft positions plus

a target bonus token. Thus `w10cap8` runs a width-10 forward but verifies only the capacity-8 prefix. In our serving implementation, `num_speculative_tokens`$= N$ corresponds to $w = N+1$. For a uniform batch of size B, the drafter processes $M = Bw$ rows; the target's dispatched shape also depends on padding and the per-request capacities. Appendix A gives the mappings.

Let $A_\phi(w, k)$ denote accepted draft tokens per verification block, averaged first within each prompt and then equally across prompts, for drafter parameters ϕ. Comparing $A_\phi(w_1, k)$ and $A_\phi(w_2, k)$ at fixed k measures the effect of forward layout with the acceptance ceiling held constant. In a non-causal block, additional mask positions can change predictions at earlier positions. We test whether training at a chosen layout improves predictions within the retained prefix.

The associated execution cost includes drafting, verification, and other work:

$$T_{\text{round}}(w, \mathbf{k}) = D(w, B, \ell) + V(s(\mathbf{k}), \ell) + H(w, \mathbf{k}, \ell), \tag{1}$$

where ℓ is the context regime and $s(\mathbf{k})$ is the executed verification shape. Acceptance and execution cost jointly determine useful widths. We use measured costs to screen configurations, then evaluate delivered tokens per unit time in the serving engine under continuous batching.

2.2 Choosing a Training Layout

Execution measurements motivate width 10 as the narrow training layout in our main study: at batch 16 on MI308X, it reduces measured draft-component time from 8.642 ms at width 9 to 7.213 ms. We therefore study training at width 10 alongside original-width training at width 16, using standard cross-entropy. Let w specify the forward layout and s the number of supervised proposal positions. For prefix x, target features h, target-generated continuation y, and anchor-plus-mask block z_w, the objective is

$$\mathcal{L}(\phi; w, s) = \mathbb{E}_{(x,h,y)} \left[-\frac{1}{s} \sum_{j=1}^{s} \log q_\phi(y_j \mid x, h, z_w) \right], \qquad s \le w - 1. \tag{2}$$

The controlled study uses $(w, s) \in \{(16, 15), (16, 9), (10, 9)\}$. The width-16 and width-10 heads with $s = 9$ isolate training forward width while fixing the supervised prefix and its loss normalization. The third head measures the effect of supervising all positions at width 16. All three share initialization, training sequence order, block starts, and 500 optimizer updates within each seed. This design measures how the forward layout changes learning under the same supervised examples and optimization schedule. We evaluate the resulting acceptance profiles offline, then deploy the same matched heads online. Appendix C reports the original full-supervision recipe and mixed-width extensions.

2.3 Choosing a Serving Configuration

Training changes $A_\phi(w, k)$ across evaluation widths, so the useful deployment point must be selected on the trained head. We measure serving throughput for each candidate width and compare the adapted head with original-width continuation after giving both the same selection opportunity. The training and serving widths need not coincide: all six matched heads select width 8, and the width-10-trained heads retain their acceptance advantage at that narrower deployment layout.

The main online candidate set is $\{8, 10, 16\}$: a narrower serving option, the adapted training width, and the original training width. All three are measured for both heads on calibration prompts. For each head, we freeze the width with the highest mean throughput before evaluation. The original training width remains available to both heads. A separate joint-deployment study also varies verification allocation (Appendix D).

3 Controls and Evaluation Design

The experiments answer three questions in sequence:

1. **Does layout affect retained predictions?** Vary inference width while fixing verification capacity (Section 4).

2. **Can training change that dependence?** Vary training width while matching supervised positions, data, and optimizer updates (Section 5.1).
3. **Does the changed profile help serving?** Let both matched heads select their deployment widths, then compare them on the frozen evaluation workload (Section 6.1).

Models, hardware, and head controls. The main target is Qwen3-8B with a five-layer DFlash-family drafter already adapted to the evaluation domain. We compare narrow-layout training (C_m, width 10) with wide-layout training (P_m, width 16), both supervising nine positions. A full-supervision width-16 head (B_m) completes the three-arm control. Table 1 summarizes the arms. Serving uses adapted vLLM (Kwon et al., 2023) on MI308X; paired runs share the runtime tree. Public Qwen3 checkpoints, a CUDA platform, and other targets provide additional tests (Section 7; Appendix A.1).

Data and workloads. The main training recipe uses 5,313 target-generated training sequences and 252 held-out sequences, AdamW, learning rate 10^{-4}, and cosine decay with warmup. Initial offline controls use 48 prompts across six task families. A frozen additional set contains 900 prompts from six sources for three-seed replication and supervision-matched training controls. The main matched-head serving test uses the 900-prompt set for three consecutive passes, batch ceiling 16, output cap 512, TRITON attention, and disabled prefix caching. Earlier serving controls use 72 prompts with warmed prefix caching. The "2k" setting denotes the model-length limit. External checks use MT-Bench (64 prompts), MATH-500 (30), and ShareGPT (24). Appendix B documents the training sources and exact-text overlap checks, including the disjoint GSM8K test-split subsets used for distillation and evaluation.

Generation and verification. The main serving comparison uses greedy generation (temperature 0) and the engine's target-verification path. Both selected heads use a fixed width-8 configuration with seven draft candidates; adaptive verification is disabled. The target model, verifier, and stopping settings are shared, with an output cap of 512 tokens. Adaptation changes the drafter weights. The separate joint-deployment experiment also changes verification allocation and is reported in Appendix D.1.

Comparisons and measurements. The training study measures mean accepted drafts per block, A_ϕ, on the same prompts and evaluation layouts. Its bootstrap resamples shared prompt IDs conditional on the three training seeds; across-seed variability is reported separately. The initial frozen-head check reports pooled committed-per-forward (CPF), which includes delivered bonus tokens and weights prompts by their block counts. Appendix A gives both definitions. Online evaluation uses delivered-token throughput: ratios are computed within pairs, averaged within each seed, and then averaged equally across seeds. Appendix M links the comparisons to archived records and reconstruction scripts.

4 Does Forward Layout Change Acceptance?

The frozen deployed head reaches 3.6146 CPF at width 8, but 3.7337 with a width-17 forward retaining the same capacity-8 prefix. The narrower forward therefore incurs a 3.19% penalty without changing the verification ceiling. Five of six task families have the same direction (Appendix B, Table 5).

We next train at the narrower layouts and measure how this gap changes.

5 How Training Reshapes the Acceptance Profile

5.1 Separating Forward Width from Supervision

Training forward width reshapes acceptance with identical supervision. We train the three arms in Table 1 from the same initialization, sequence order, block starts, and 500-update budget within each seed. $C_m - P_m$ changes forward width with nine supervised positions and their loss normalization fixed; $P_m - B_m$ changes supervision range at width 16.

Table 1: **Forward-width adaptation with supervision held fixed.** Mean per-prompt accepted drafts per block, averaged over 900 prompts and three training seeds. All three evaluation layouts use capacity 8. The subscript m denotes newly trained heads with matched block starts.

Training arm	w	s	`w8`	`w10cap8`	`w17cap8`
B_m: original-width	16	15	2.5508	2.5752	2.6581
P_m: prefix-supervised	16	9	2.5515	2.5777	2.6570
C_m: narrow-layout	10	9	2.6007	2.6351	2.6373

At `w10cap8`, $C_m - P_m$ is $+0.0574$; at `w17cap8`, it is -0.0196 (Table 1). The layout interaction is $+0.0770$, with prompt-bootstrap interval $[0.0641, 0.0903]$ and positive effects in all three seeds. Supervision-range contrasts are much smaller ($+0.0025/-0.0012$), with intervals containing zero. Thus concentrating the loss on early positions does not explain the observed layout interaction: it persists when those positions are identical. Appendix B.2 gives the training controls, seed results, and conditional bootstrap protocol.

5.2 The Gain Extends to the Deployed Width

The narrow-trained head retains its advantage at width 8, where both heads are deployed (Figure 1a). Here $C_m - P_m$ is $+0.0493$ accepted drafts per block, with prompt-bootstrap interval $[0.0389, 0.0598]$. All three seed contrasts are positive ($+0.0494, +0.0465, +0.0519$), as are all six source means and their prompt-bootstrap intervals. The gain thus extends beyond the width-10 training layout.

Across widths 8, 10, and 17, the wide-trained controls B_m and P_m have nearly overlapping acceptance curves. Narrow-layout training raises the narrow end of the curve while giving up some acceptance at width 17. This suggests a deployment strategy: adapt the drafter to improve predictions at efficient narrow layouts, then reselect its serving width. Appendix B.3 gives the deployed-width breakdown.

6 When Does the Changed Profile Improve Serving?

6.1 Matched Heads Improve Throughput After Width Selection

The acceptance advantage at width 8 carries through to serving with the same target and verification configuration. Each P_m and C_m head independently chooses from $\{8, 10, 16\}$ on calibration prompts, and its choice is frozen before evaluation. All six seed–head combinations select width 8, including the heads trained at width 10. The subsequent test uses the 900-prompt workload for three consecutive passes with prefix caching disabled, yielding 2,700 requests and 9.1–9.3 minutes of measured serving per run. Each of three sessions contains all three seeds and two reversed-order pairs per seed; both arms of a pair run sequentially on the same device.

Table 2: **Online gains from training layout with supervision matched.** Both heads independently select width 8. Each seed averages six paired comparisons across three sessions. CPF differences are absolute logged-CPF increments; throughput changes are relative.

Training seed	Decode gain	Wall gain	CPF $C_m - P_m$
0	$+1.54\%$	$+1.54\%$	$+0.0542$
1	$+1.52\%$	$+1.52\%$	$+0.0484$
2	$+1.67\%$	$+1.67\%$	$+0.0449$

Averaging equally across seeds gives a 1.58% throughput gain, positive in all three seeds and sessions and in 17 of 18 pairs (Table 2). Both heads have already selected width 8: the gain comes from the changed drafter at that shared configuration. Training therefore improves the acceptance profile available to width selection, and the improvement carries through to serving (Figure 1b). Appendix D.2 reports the paired effects, uncertainty summaries, and checks across run order, sessions, and devices.

Complementary deployment evidence. Combining adaptation with forward-width and verification-allocation choices yields a 5.6% gain over the faster frozen static-width reference in a separate study. This is a complete-configuration gain using the original full-supervision C head. Appendix D.1 gives the configuration comparison and its static-width baselines.

7 Deployment Conditions for Adaptation

Models and workloads. Adaptation gains vary across models and workloads. Public Qwen3-8B and Qwen3-4B checkpoints show matched-layout gains beyond equal-update continuation (Table 6). Llama-3.1-8B shows no detected frozen-head narrow-layout penalty in its fixed-capacity test; its specialization adds little throughput on MI308X and reduces it by 1.45% in the CUDA platform's selected-width comparison. The 27B/DFlash2 training increment is $+0.49\% \pm 0.74\%$. On MT-Bench, the original training recipe's matched-width throughput changes are $+3.03/-0.83/+0.45\%$ across three seeds, despite positive acceptance interactions. For deployment, these results suggest measuring the retained-prefix profile first, comparing adaptation against equal-update continuation, and letting both heads reselect their widths (Appendices F–H).

Execution costs. The MI308X draft profile falls from 8.642 to 7.213 ms between widths 9 and 10; isolated four-GEMM layer sums fall from 933.2 to 486.6 μs. The CUDA platform's corresponding layer sums are nearly equal, 507.5 and 509.5 μs. Hardware therefore changes which widths are worth considering. Static 27B, quantized-target, and Mac mini configuration studies extend the cost measurements to other execution paths (Appendix H).

When speculation is worth enabling. The main gain is measured at batch ceiling 16 with short inputs. In a separate concurrency scan, the best tested speculative arm reaches $1.293\times$ no-speculation throughput at ceiling 16, falling to $0.847\times$ at 32 and $0.677\times$ at 64. A same-width head comparison still improves the speculative path at ceiling 32, but ordinary decoding is faster in the scan. At 30k context, even the best tested configuration reaches only $0.497\times$ no-speculation throughput. These are settings where the deployment choice is to disable speculation; full measurements appear in Appendices H.1 and I.

8 Related Work

Training layout and width selection. DFlash evaluates crossed training and inference block sizes and reports asymmetric cross-width generalization; it also studies position-dependent loss decay that emphasizes early draft tokens (Chen et al., 2026). Adaptive Block Diffusion trains over prefix–window configurations to improve cross-configuration denoising, evaluated through language-model perplexity (Jain, 2026). We separately control verification capacity and training supervision to identify how forward layout affects retained-prefix acceptance. The resulting training-layout interaction persists with supervision matched, and we deploy the same heads after independent width selection. BlockPilot instead learns an instance-adaptive inference-width policy (Zhang et al., 2026). A width policy can also select configurations for an adapted head. Appendix L provides contextual replay of released acceptance artifacts.

Speculative allocation and execution costs. DSpark combines confidence-scheduled verification with hardware profiles (Cheng et al., 2026). Adaptive tree and length methods likewise change the speculative work executed (Li et al., 2024; Huang et al., 2024; Hou et al., 2025). Such allocators can consume an adapted drafter. Profiling, batching, and long-context inference research explain why accepted tokens must be considered alongside execution cost (Agrawal et al., 2024; Su et al., 2023; Sadhukhan et al., 2025). Our measurements connect these costs to model adaptation and complete serving configurations.

9 Limitations

The supervision-matched evidence comes from Qwen3-8B. Other targets and workloads show smaller or mixed training gains, so the useful adaptation layout must be evaluated for each deployment. The experiments identify a behavioral effect of forward layout; the internal network

mechanism remains open. Matching supervised positions and optimizer updates also leaves forward FLOPs different.

The main serving test spans three seeds and sessions, using nine-minute runs on a repeated 900-prompt workload. Production arrival patterns and longer service periods require further evaluation. Offline bootstrap intervals are conditional on the three seeds; the online appendix reports seed, session, and device variation separately. Integration with learned width policies such as BlockPilot remains to be tested. The released supplement supports numerical reconstruction; exact-environment reproduction is limited by the hardware and software disclosures in Appendix A.1.

10 Conclusion

Training forward layout changes retained-prefix acceptance even when supervised positions and verification capacity are fixed. Across three seeds and 900 prompts, narrow-layout training improves acceptance at widths 8 and 10 while lowering it at width 17. The gain persists at the selected deployment width, where the same heads deliver 1.58% higher throughput after independent configuration selection. These results connect model adaptation with width selection: measured costs guide the layouts we consider, and training improves the predictions available at deployment.

AI USE STATEMENT

Generative AI tools assisted with refining hypotheses and the conceptual framing, providing feedback on experimental design, implementing and debugging experimental and analysis code, interpreting results, and drafting and revising the manuscript and figures. Target-model-generated continuations were also used as distillation supervision, as described in Appendix B. Checks on AI-assisted artifacts included code execution and smoke tests, reconstruction of numerical summaries from archived run outputs, and cross-checks between reported values and their source records. These checks do not replace scientific judgment; the authors are responsible for the final text, claims, code, and experimental interpretation.

REPRODUCIBILITY STATEMENT

Section 3 and Appendices A–C describe configurations, data, metrics, and training controls; the subsequent appendices report additional results and protocol-specific limitations. Appendix M describes the internal numerical reconstruction archive. The separate numerical supplement supplies field-selected records from 350 serving runs and thirty-eight offline probes, with standard-library Python scripts to reconstruct the principal training, layout, and deployment contrasts. Its README specifies coverage and exclusions; it supports numerical verification, not full training or online serving reproduction. The accompanying research-code archive provides training and evaluation implementations, local runtime patches, CPU tests, and plotting programs. An anonymous online code repository is available at https://anonymous.4open.science/r/cost_llm_with_tile-7E79/. The online repository contains code and setup documentation; measurement extracts and manuscript materials are provided in the accompanying supplement. The manuscript source package separately contains LaTeX sources and figures. Exact-environment replication is also limited by the device and software-build disclosure restrictions stated in Appendix A.1.

REFERENCES

Amey Agrawal, Nitin Kedia, Jayashree Mohan, Ashish Panwar, Nipun Kwatra, Bhargav Gulavani, Ramachandran Ramjee, and Alexey Tumanov. Vidur: A Large-Scale Simulation Framework For LLM Inference. In *Proceedings of Machine Learning and Systems (MLSys)*, 2024. URL https://arxiv.org/abs/2405.05465.

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. URL https://arxiv.org/abs/2302.01318.

Jian Chen, Yesheng Liang, and Zhijian Liu. DFlash: Block Diffusion for Flash Speculative Decoding. *arXiv preprint arXiv:2602.06036*, 2026. URL https://arxiv.org/abs/2602.06036.

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. URL https://arxiv.org/abs/2607.05147.

Yunlong Hou, Fengzhuo Zhang, Cunxiao Du, Xuan Zhang, Jiachun Pan, Tianyu Pang, Chao Du, Vincent Y. F. Tan, and Zhuoran Yang. BanditSpec: Adaptive Speculative Decoding via Bandit Algorithms. *arXiv preprint arXiv:2505.15141*, 2025. URL https://arxiv.org/abs/2505.15141.

Kaixuan Huang, Xudong Guo, and Mengdi Wang. SpecDec++: Boosting Speculative Decoding via Adaptive Candidate Lengths. *arXiv preprint arXiv:2405.19715*, 2024. URL https://arxiv.org/abs/2405.19715.

Gagan Jain. Adaptive Block Diffusion: Resolving Training-Inference Mismatch in Diffusion Language Models. *arXiv preprint arXiv:2606.29275*, 2026. URL https://arxiv.org/abs/2606.29275.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Symposium on Operating Systems Principles (SOSP)*, 2023.

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

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. EAGLE-2: Faster Inference of Language Models with Dynamic Draft Trees. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024. URL https://arxiv.org/abs/2406.16858.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. YaRN: Efficient Context Window Extension of Large Language Models. *arXiv preprint arXiv:2309.00071*, 2023. URL https://arxiv.org/abs/2309.00071.

Ranajoy Sadhukhan, Jian Chen, Zhuoming Chen, Vashisth Tiwari, Ruihang Lai, Jinyuan Shi, Ian En-Hsu Yen, Avner May, Tianqi Chen, and Beidi Chen. MagicDec: Breaking the Latency-Throughput Tradeoff for Long Context Generation with Speculative Decoding. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/13f972adf12bdf886583d48cd528002f-Paper-Conference.pdf.

Qidong Su, Christina Giannoula, and Gennady Pekhimenko. The Synergy of Speculative Decoding and Batching in Serving Large Language Models. *arXiv preprint arXiv:2310.18813*, 2023. URL https://arxiv.org/abs/2310.18813.

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. URL https://arxiv.org/abs/2606.31315.

A CONFIGURATION AND METRIC CONVENTIONS

Table 3: Width and treatment identities used in this paper. A training width is not an engine token-count setting.

Notation / arm	Meaning
`w8`	Offline width 8: one anchor and seven proposal slots, capacity 8.
`w17cap8`	Offline width 17, retaining the same capacity-8 verification prefix.
`w10cap8`	Offline width 10, retaining capacity 8.
R2	Width-8 CE specialist, 2,000 microsteps (500 updates).
R3	Width-10 CE specialist, 2,000 microsteps (500 updates); used in the original B/C and joint-deployment studies.
W-MIX	Uniform step-level mixture of widths 4, 8, 10, 16; 2,000 total microsteps (500 updates); source of long-context results.
Live baseline	$N = 8$, query width 9, batch-16 draft rows 144.
Live adapted	$N = 9$, query width 10, batch-16 draft rows 160.
Live capacity set	B″: observed nonzero capacities $\{2, 4, 8\}$; P1/L2: grid configuration recorded explicitly.
L2 fixed references	W8: $N = 7$, query width 8; W16: $N = 15$, query width 16, both frozen full-verification.

The original checkpoint's training label "block size 16" is recorded separately from every experiment's explicit query width. We do not identify an offline width-8 evaluation with an engine $N = 8$ configuration. Offline CPF comparisons use the same harness within each experiment; live effects are evaluated separately.

Acceptance measures and their relationship. For prompt i, let m_i be its number of verification blocks, r_{ij} the accepted draft tokens in block j, and q_{ij} the delivered committed tokens, including any target bonus and final truncation. The main training analysis uses

$$A_\phi(w,k) = \frac{1}{n}\sum_{i=1}^{n}\frac{\sum_{j=1}^{m_i} r_{ij}}{m_i}, \qquad \mathrm{CPF}_{\mathrm{pooled}} = \frac{\sum_i \sum_{j=1}^{m_i} q_{ij}}{\sum_i m_i}. \tag{3}$$

The first quantity gives equal weight to prompts and counts accepted drafts; the second weights by verification blocks and counts delivered tokens. They are distinct summaries, not interchangeable absolute scales. Both are computed within their stated workload and protocol.

Offline pooled CPF divides delivered committed tokens by verification opportunities, including final truncation. Live benchmark CPF instead uses the logged acceptance count: $(\texttt{accepted} + \texttt{steps})/\texttt{steps}$, where steps are summed per-request verification steps. This acceptance statistic can differ from delivered output tokens divided by steps; all throughput measures use delivered tokens. In serving logs, summed per-request target steps may differ from the number of engine batch forwards. A reduction in those steps therefore does not by itself quantify GPU time saved. When a report uses median TPOT multiplied by pooled CPF, we treat it as a diagnostic quantity, not an exactly measured round-time mean.

Original recipe and checkpoint names. The earlier studies denote the frozen deployed head by A, original-width continuation by B, width-10 continuation by C, and mixed-width continuation by D. B/C/D share initialization and data. These heads differ from the newly trained supervision-matched heads used in the main result.

The deployment recipe uses $s = w - 1$: width-10 continuation supervises positions 1–9, while width-16 continuation supervises positions 1–15. Our three-arm control separates these choices by training $(w, s) \in \{(16, 15), (16, 9), (10, 9)\}$ with common block starts (Section 5.1). A specialist fixes w; a mixed-width head samples w at each microstep. Measured costs guide the candidate layouts. The serving C_m/P_m comparison evaluates the supervision-matched intervention; historical B/C comparisons evaluate the original complete recipe.

Our width-10 specialist is denoted R3, the width-8 specialist R2, and the uniform mixture over $\{4, 8, 10, 16\}$ by W-MIX. Each receives 2,000 microsteps with accumulation four, or 500 optimizer

updates. On the recorded MI308X training runs, original-width continuation, R3, and W-MIX take 215.4, 205.6, and 199.9 seconds, respectively, excluding data generation, loading, and deployment profiling. Equal-update controls match data and optimization steps, not supervised token count or FLOPs.

A.1 GPU PLATFORMS AND EXECUTION IMPLEMENTATION

The CUDA platform's device model and identifying nominal specifications are withheld under institutional disclosure requirements. Table 4 reports the main execution settings; Section 7 reports the measured draft-component and isolated GEMM costs. All latency and throughput results are measured on the actual device. Withholding the device identity and exact software build limits exact-environment replication.

Table 4: CUDA platform: main short-input serving configuration. Extension experiments retain their separately stated settings.

Property	Configuration
GPU execution	One GPU per inference run (tensor parallelism 1); device-balanced slots where specified
Runtime base	CUDA 12.8; adapted vLLM; paired arms share the same runtime tree
Main model precision	BF16 target and drafter; AWQ target experiments reported separately
Main attention backend	`TRITON_ATTN`; the 30k backend comparison additionally uses FLASH attention
Short-input configuration	Batch ceiling 16, model-length limit 2,048, prefix caching enabled
Head-only deployment test	72 prompts, output cap 512, engine draft count 9 (query width 10); 16 pairs (32 runs)

Execution shapes and implementation paths. An engine draft count N corresponds to query width $N + 1$; batch size b gives $M = b(N + 1)$ drafting rows. Verification capacity is configured separately. The isolated BF16 measurements time the four draft-layer matrix multiplications with device events, whereas the integrated draft profile includes the complete draft component. Their absolute times are not interchangeable. The main width-10 head controls hold the forward configuration fixed; adaptive verification can still respond to changed drafter predictions. The AWQ tests on the CUDA platform use a specialized weight-only quantized matrix-multiplication path and native DSpark budget allocation, while the ROCm AWQ tests use a Triton path and threshold allocation.

Graph coverage and timing boundaries. The native-allocator diagnostics use a full-graph capture limit of 128 target tokens. The verify-table index counts non-draft tokens plus the draft budget, so profiling must include reachable sizes beyond that limit. Those diagnostics align verification timing to `forward_start–forward_end`, excluding logits and sampling, and match the boot profiler's context length of 8,192. These are the native-table diagnostic settings, not a claim that every experiment shares one context or capture configuration. Measured component costs, request-step counts, and episode throughput retain their distinct definitions.

Serving measurement. Decode throughput uses delivered output tokens over the union decode window; benchmark-wall throughput uses the benchmark wall interval. Pairwise relative changes use the same timing field within each comparison. Synchronizing timing ledgers and diagnostic hooks are disabled in formal throughput runs. The native-allocator clock-control precheck did not reduce observed variability, and its formal comparisons use default clocks; that precheck does not establish a fixed noise threshold. Per-experiment pairing, device assignment, output caps, and repeat counts are reported with the corresponding results.

B Training Data and Common-Prefix Controls

Table 5: Offline common-prefix controls, 48 prompts. Every row in the upper block uses verification capacity 8. Lower rows examine the narrow–wide gap within the same adapted checkpoint. All values are pooled CPF.

Checkpoint	Draft/verification configuration	CPF
Deployed	`w8`	3.6146
Deployed	`w10cap8`	3.6656
Deployed	`w17cap8`	3.7337
R3, width-10 CE	`w10cap8`	3.7369
R2, width-8 CE	`w8`	3.7149
R2, width-8 CE	`w17cap8`	3.7211

The adaptation pool contains 5,313 training and 252 held-out target-generated sequences. An exact-message check between the 6,239-message source pool and the 161-message evaluation pool finds no overlap. The GSM8K contribution to the source pool comprises 1,200 messages from the official test split, disjoint from the 20 GSM8K evaluation items. Labels are the target's own greedy continuations, not reference answers. We disclose this source choice rather than describing the protocol as a conventional official train/test split. Of 161 evaluation prompts, 80 are public-source chat, HumanEval, and GSM8K items verified against their archived sources; 81 are self-built. Exact-text separation does not rule out semantic or pretraining overlap. The external-workload tests supplement this domain-proximate evaluation.

Training layout and supervision. Equation 2 distinguishes forward width w from supervised prefix length s. In the original B/C recipe, $s = w - 1$ and loss is averaged over the selected positions across packed blocks. Valid block starts also depend on the training width. The additional three-arm experiment below matches block starts across newly trained heads and compares widths 10 and 16 under identical nine-position supervision. These heads are denoted $B_m/P_m/C_m$ to distinguish them from the original serving checkpoints.

The Qwen3-4B replication generates its own 2,500 target continuations (420,760 tokens), rather than reusing 8B labels. All matched B/C controls share initialization and data and use terminal checkpoints after 500 updates. The original A/B/C heads use the recorded weight-prefix fingerprints A: `cf982649`, B: `c9ac2d06`, C: `d36f6071`. These prefix fingerprints are distinguished from full-file hashes.

Table 6: Equal-step controls on public checkpoints. Both CE arms use 2,000 microsteps (500 updates) and the same target-specific training data. CPF is evaluated at the indicated fixed layout. The final two rows report relative contrasts; the 4B original-width control is taken directly from its archived evaluation JSON.

Target	Training arm	`w8`	`w10cap8`	`w16`
8B	Public checkpoint	3.3538	3.3899	3.8476
8B	Original-width CE (16)	3.4725	3.5223	4.0264
8B	Deployment-width CE (10)	3.5564	3.5679	4.0449
4B	Public checkpoint	3.1693	3.1891	3.6474
4B	Original-width CE (16)	3.2706	3.2893	3.7391
4B	Deployment-width CE (10)	3.3467	3.4136	3.8007
8B	Width-10 vs width-16 CE	$+2.41\%$	$+1.30\%$	$+0.46\%$
4B	Width-10 vs width-16 CE	$+2.33\%$	$+3.78\%$	$+1.65\%$

The frozen narrow-forward CPF penalty is 3.19% in the pooled statistic. The corresponding mean per-prompt difference is -4.09%, with the archived prompt-bootstrap interval $[-6.2, -2.1]\%$. These weightings answer different aggregation questions. The public-checkpoint table compares identical layouts within each column; unrestricted columns have different verification ceilings.

B.1 A Frozen 900-Prompt Evaluation Set

We construct 900 additional prompts before evaluation: 200 Chinese instructions from Alpaca-GPT4-zh, 200 first human turns from ShareGPT, 104 HumanEval problems (IDs 60–163), 96 MBPP tasks, 150 MATH-500 problems, and 150 IFEval prompts. Source quotas and the manifest are fixed before model evaluation. The construction excludes normalized-text matches with the training source pool and prior evaluation lists, checks wrapped and raw MBPP forms, and records a content hash for every prompt. GSM8K is excluded from this extension. The archived source verifier also checks source formatting and deterministic reconstruction.

The original B/C heads from seeds 0, 1, and 2 are evaluated at `w8`, `w10cap8`, and `w17cap8`, with greedy decoding and a 128-token output cap. Each cell contains all 900 prompt IDs. Let $a_{h,r,i}(l)$ be accepted draft tokens divided by verification blocks for prompt i, head h, seed r, and layout l. We average the paired prompt differences within each seed, then average the three seed estimates. This is a mean of per-prompt acceptance rates, distinct from the pooled delivered CPF of the original 48-prompt tables.

The primary interaction is $\Delta_2 = (C - B)_{\texttt{w10cap8}} - (C - B)_{\texttt{w17cap8}}$. Its seed estimates are $+0.0757, +0.0755, +0.0702$, giving mean $+0.0738$ and across-seed SE 0.0018. We draw 10,000 bootstrap replicates, resampling the same prompt indices across all cells and seeds (random seed 1561). The resulting conditional interval is $[0.0617, 0.0859]$. Source-level intervals are positive for all six sources (Table 9). The secondary interaction using `w8` instead of `w10cap8` is $+0.0670$, interval $[0.0543, 0.0798]$.

For the original 48 prompts, the retrospective mean interaction is $+0.174$ accepted draft tokens per block, interval $[0.076, 0.289]$; delivered-CPF sensitivity gives $+0.169$, interval $[0.073, 0.281]$. The two prompt sets remain separate analyses.

B.2 Matching Supervision and Block Starts During Training

All three arms start from the same domain-adapted initialization and are retrained for seeds 0, 1, and 2. B_m has $(w, s) = (16, 15)$, P_m has $(16, 9)$, and C_m has $(10, 9)$. A common width-16 validity constraint selects block starts for all arms. With the same sequence order and deterministic start selection, this holds the examples and start locations fixed within each seed. P_m retains all 16 forward positions and their computation paths; only the first nine proposal positions contribute direct cross-entropy terms. P_m/C_m average loss over the same nine positions per block.

All nine runs use AdamW, learning rate 10^{-4}, 100 warmup microsteps, 2,000 microsteps with accumulation four, maximum sequence length 1024, at most 24 blocks per sequence, and minimum completion length 48. Each archived log contains all 2,000 steps and matches the prescribed arm configuration. The initial evaluation uses the same 900 prompts at widths 10 and 17 with capacity 8. The width-8 follow-up completes a three-layout profile: 27 cells across three arms and three seeds, each containing all 900 prompts. Table 7 reports the individual seed means.

Table 7: Supervision-matched training: mean per-prompt accepted draft tokens per block for each seed. Each entry averages all 900 prompts at the stated fixed-capacity layout.

Arm	Seed	`w8`	`w10cap8`	`w17cap8`
B_m	0	2.5537	2.5754	2.6601
B_m	1	2.5503	2.5768	2.6563
B_m	2	2.5483	2.5736	2.6580
P_m	0	2.5519	2.5770	2.6563
P_m	1	2.5532	2.5797	2.6555
P_m	2	2.5493	2.5765	2.6590
C_m	0	2.6014	2.6341	2.6416
C_m	1	2.5997	2.6329	2.6339
C_m	2	2.6011	2.6382	2.6364

Define $F_l = (C_m - P_m)_l$ and $S_l = (P_m - B_m)_l$. The primary contrast is $\Delta_F = F_{\texttt{w10cap8}} - F_{\texttt{w17cap8}}$; its seed estimates are $+0.0718, +0.0748, +0.0843$. We use 10,000 shared-prompt bootstrap repli-

cates (random seed 1562), conditional on these three training seeds. Across-seed SE and prompt-bootstrap intervals describe different sources of variation and are reported separately in Table 8. The matching experiment was specified before training, including both possible primary-result directions.

Table 8: Matched-training contrasts. Units are accepted draft tokens per block. SE is across three seed means; intervals use shared-prompt resampling conditional on those seeds.

Contrast	Mean	Seed SE	Prompt-bootstrap 95% interval
$F_{\texttt{w10cap8}}$	$+0.0574$	0.0025	$[+0.0469, +0.0680]$
$F_{\texttt{w17cap8}}$	-0.0196	0.0025	$[-0.0270, -0.0121]$
Δ_F	$+0.0770$	0.0038	$[+0.0641, +0.0903]$
$S_{\texttt{w10cap8}}$	$+0.0025$	0.0004	$[-0.0018, +0.0066]$
$S_{\texttt{w17cap8}}$	-0.0012	0.0014	$[-0.0052, +0.0027]$
Δ_S	$+0.0036$	0.0010	$[-0.0016, +0.0090]$
Matched $(C_m - B_m)$ interaction	$+0.0806$	0.0028	$[+0.0671, +0.0944]$

The $C_m - P_m$ contrast changes sign between layouts in every seed. The matched $C_m - B_m$ interaction is $+0.0806$, close to the original-head interaction on the same prompts ($+0.0738$). Supervision-range estimates are small and their prompt intervals contain zero. All six source-level Δ_F estimates are positive; five intervals exclude zero, while HumanEval has the widest uncertainty relative to its effect.

Table 9: All sources in the 900-prompt evaluation. Each cell gives the mean interaction and conditional prompt-bootstrap 95% interval. Original Δ_2 uses the existing B/C checkpoints; matched Δ_F compares newly trained C_m/P_m with identical supervision.

Source	Prompts	Original Δ_2 [interval]	Matched Δ_F [interval]
Chinese instructions	200	$+0.0773$ $[+0.0563, +0.0990]$	$+0.0846$ $[+0.0648, +0.1059]$
ShareGPT	200	$+0.0771$ $[+0.0602, +0.0943]$	$+0.0800$ $[+0.0604, +0.1027]$
HumanEval	104	$+0.0515$ $[+0.0028, +0.1001]$	$+0.0403$ $[-0.0055, +0.0879]$
MBPP	96	$+0.0610$ $[+0.0048, +0.1206]$	$+0.0694$ $[+0.0046, +0.1392]$
MATH-500	150	$+0.1028$ $[+0.0685, +0.1371]$	$+0.1147$ $[+0.0783, +0.1507]$
IFEval	150	$+0.0595$ $[+0.0419, +0.0785]$	$+0.0553$ $[+0.0371, +0.0752]$

B.3 Acceptance at the Selected Deployment Width

We evaluate all nine matched-control heads at width 8 on the same frozen 900-prompt set, using the offline harness and 128-token output cap of the width-10/17 evaluation. Capacity 8 allows seven draft candidates and a target bonus token. This follow-up tests the offline acceptance contrast at the width already selected by the online calibration; it uses the existing evaluation set without further training or prompt selection.

The three $C_m - P_m$ seed contrasts are $+0.0494, +0.0465, +0.0519$, averaging $+0.0493$ with seed SE 0.0016. Shared-prompt resampling (10,000 draws, random seed 1563), conditional on the three seeds, gives interval $[0.0389, 0.0598]$. Source-level results appear in Table 10. The secondary $B_m - P_m$ contrast is -0.0007, interval $[-0.0044, 0.0031]$, consistent with the small supervision-range contrasts at the other two layouts. The offline result and the online throughput gain assess the same heads at width 8; their magnitudes use different metrics and are reported separately.

Table 10: Matched-head acceptance at the selected width 8. Each contrast is $C_m - P_m$ in mean accepted draft tokens per block. Intervals resample shared prompts conditional on three training seeds.

Source	Prompts	Mean contrast	Prompt-bootstrap 95% interval
Chinese instructions	200	$+0.0609$	$[+0.0433, +0.0801]$
ShareGPT	200	$+0.0378$	$[+0.0232, +0.0542]$
HumanEval	104	$+0.0419$	$[+0.0064, +0.0837]$
MBPP	96	$+0.0504$	$[+0.0108, +0.0911]$
MATH-500	150	$+0.0719$	$[+0.0410, +0.1025]$
IFEval	150	$+0.0307$	$[+0.0108, +0.0488]$
All sources	900	$+0.0493$	$[+0.0389, +0.0598]$

C Specialists, Mixed Widths, and Live Training Controls

C.1 Earlier Training Controls at a Fixed Forward Width

Layout adaptation improves acceptance beyond equal-update continuation in all three MI308X serving sessions (Table 11). Both heads run at width 10 with identical serving settings; only the checkpoint changes. Throughput gains range from 0.87% to 5.48% across the sessions. The expanded session contributes 16 pairs, with 13 positive throughput effects. A uniform decode-window reconstruction gives a descriptive inverse-variance mean of $3.09\% \pm 0.58\%$. Table 11 reports the individual sessions alongside this summary.

Table 11: **Training adaptation beyond continued training (C/B).** Identical width-10 serving settings, fixed seed-0 heads. Throughput uses the decode window throughout. SEs describe within-session pairs; the pooled row is descriptive.

Device	Session	Pairs	Throughput change	CPF change
MI308X	Four-head	4	$+5.48\% \pm 1.16$	$+2.68\% \pm 0.39$
MI308X	Independent repeat	4	$+0.87\% \pm 1.47$	$+1.28\% \pm 0.65$
MI308X	Expanded repeat	16	$+2.66\% \pm 0.75$	$+2.84\% \pm 0.45$
MI308X	Descriptive summary	24	$+3.09\% \pm 0.58$	—
CUDA platform	Equal-update control	4	$+2.68\% \pm 2.18$	$+3.31\% \pm 0.58$

The CUDA platform supplies a complementary head-only comparison. At width 10, C improves decode throughput over the original deployed head A by $4.03\% \pm 1.23\%$ across 16 pairs (12 positive; SE across pairs), and CPF by $2.88\% \pm 0.33\%$ (all positive). The equal-update C/B control on this platform is $+2.68\% \pm 2.18\%$ (Table 11). On this platform, isolated GEMM costs at $M = 144$ and 160 are nearly equal, giving a second hardware setting for evaluating adaptation.

Table 12: Four domain heads evaluated with the same offline probe protocol (48 prompts). A is the frozen initialization; B/C/D share that initialization and data and each receive 2,000 microsteps (500 updates). CPF values are pooled within each layout, not comparable verification ceilings across columns.

Head	`w8`	`w10cap8`	`w16`
A: frozen deployed	3.6146	3.6656	4.2553
B: original-width CE (16)	3.6686	3.6839	4.2718
C: width-10 CE (R3)	3.7274	3.7369	4.2594
D: W-MIX	3.7306	3.7306	4.2390

Table 13: Same-forward four-head experiment. Entries are mean within-slot relative changes ± SE over four slots; the count is positive-throughput slots. Throughput uses the benchmark-wall `tok_per_s` field. Rows share runs and are not independent experiments.

Contrast	Throughput change	Positive slots	CPF change
C/B	$+5.47\% \pm 1.14$	4/4	$+2.68\% \pm 0.39$
D/B	$+0.97\% \pm 1.07$	2/4	$+1.24\% \pm 0.49$
D/C	$-4.26\% \pm 0.79$	0/4	$-1.40\% \pm 0.49$
B/A	$-2.39\% \pm 1.43$	0/4	$-0.36\% \pm 0.29$
C/A	$+2.96\% \pm 2.02$	3/4	$+2.31\% \pm 0.55$
D/A	$-1.40\% \pm 2.30$	1/4	$+0.87\% \pm 0.49$

Table 14: The four Latin slots, using archived benchmark-wall TPS. Contrasts are computed within a slot before averaging; arm-mean ratios are not used for the reported SEs.

Slot	A TPS	B TPS	C TPS	D TPS	C/B	D/C
1	1301.31	1286.50	1364.80	1297.91	$+6.09\%$	-4.90%
2	1347.28	1258.58	1328.93	1248.03	$+5.59\%$	-6.09%
3	1256.62	1253.69	1352.05	1303.73	$+7.85\%$	-3.57%
4	1327.90	1306.78	1337.80	1304.91	$+2.37\%$	-2.46%

The first four-head comparison uses Latin orders ABCD, BCDA, CDAB, and DABC. Slots 1–2 and 3–4 use different GPUs; each contrast is within slot. All arms use $N = 9$ (query width 10), batch ceiling 16, threshold 0.02, 72 prompts, output cap 512, model-length limit 2048, TRITON attention, and prefix caching. The table retains its original benchmark-wall timing. Its C/B estimate ($5.47\% \pm 1.14\%$) is close to the decode-window estimate ($5.48\% \pm 1.16\%$) for these same runs; they remain distinct timing definitions. Main-text Table 11 reconstructs the C/B contrasts with decode-window timing consistently across sessions.

The independent repeat uses an A/B/C rotation with four slots; the expanded C/B comparison uses 16 slots with alternating within-pair order, eight on each GPU. They evaluate the same seed-0 B/C heads, not new training seeds. The expanded median decode-throughput effect is 2.75%, with 13/16 positive effects; CPF is positive in 15/16. The original engineering rule required both a 3.2% gain and twice the within-session SE. Its archived decisions are unchanged, but 3.2% is not used in this paper as a threshold of scientific importance. Direct recomputation with a uniform timing field gives a descriptive pooled mean of 3.09% (decode) or 3.08% (wall), both rounding to 3.1%. Pooling does not remove session heterogeneity.

W-MIX samples widths once per microstep; logged counts for widths 4/8/10/16 are 516/509/484/491. Relative to the measured specialist references, its largest offline shortfall is 0.77%. At width 16 it is 0.38% below the frozen head. In live serving, D/B is $+0.97\% \pm 1.07\%$ and D/C is $-4.26\% \pm 0.79\%$, so offline coverage does not make it an interchangeable specialist. The original four-head traces include capacities $\{0, 2, 4, 8, 9\}$: matching settings is not the same as fixing realized verification allocations.

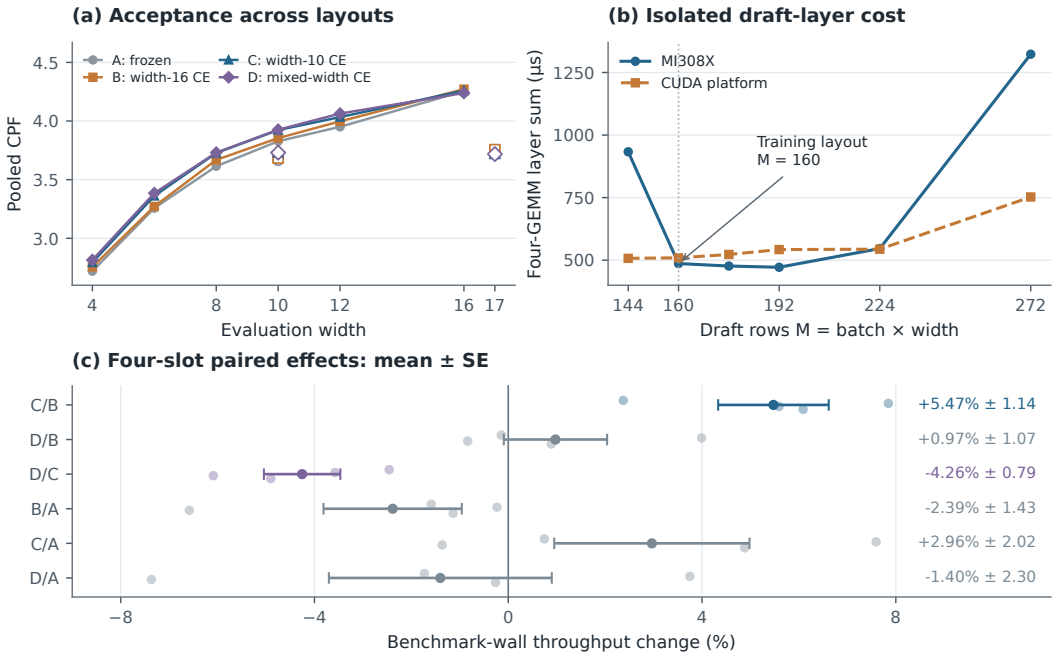

Figure 2: Earlier four-head controls. (a) Filled markers show full-verification layouts; hollow markers show capacity-8 prefixes. (b) Isolated layer costs at batch 16, with the chosen width 10 marked at $M = 160$. (c) Within-slot benchmark-wall effects: small points are the four slots and bars are mean $\pm$ SE. Panels use separate protocols. Table 11 reports the independent sessions that follow this initial C/B estimate.

D Verification Alignment and Joint Deployment

In the original MI308X joint deployment, R3 uses $N = 9$, hence $w = 10$ and $M = 160$ at batch 16. The allocator may verify fewer positions than the drafter processes. We evaluate both threshold allocation and an existing stair/fill path; forward width is fixed within a run. Stair uses frozen local width-cost anchors, and fill reads the runtime batch-cost table to extend allocations within a measured cost plateau. Both paths consume the adapted head through the same fixed-forward interface. Appendix D records the capacity controls.

D.1 Joint Deployment Adds Value Beyond Static Width Selection

The adapted configuration adds serving value beyond selecting a faster frozen width. A separate four-slot experiment compares no speculation, frozen full-verification widths 8 and 16, and the adapted joint configuration DEP (C, width 10, stair/fill, capacity grid $\{2, 4, 8\}$). Each slot rotates arm order and uses the same workload. The resulting effects appear in Table 15.

Table 15: **Joint deployment versus frozen static references.** MI308X benchmark-wall throughput, four Latin slots. W8 and W16 denote query widths 8 and 16. DEP changes the head, forward width, and verification allocation.

Contrast	Throughput change	Positive slots
DEP / faster frozen W8	$+5.55\% \pm 1.41$	4/4
DEP / frozen W16	$+55.23\% \pm 1.75$	4/4
W8 / W16	$+47.09\% \pm 0.99$	4/4
DEP / no speculation	$+26.52\% \pm 1.82$	4/4

Static W8 already improves throughput by 47.09% over W16 despite lower logged CPF (3.5849 versus 4.1143, both from L2; Appendix E). DEP adds 5.55% over W8 and is 26.52% faster than disabling speculation. The 55.23% gain combines the head, width, and verification choices; Appendix C.1 measures the training gain with the serving configuration held fixed.

Verification remains part of that choice. In separate paired experiments, changing the head and forward width with unconstrained verification improves CPF by 6.06% but throughput by only 1.80%. A fixed-capacity-8 control yields $6.75\% \pm 0.82\%$ higher throughput. The verification budget is therefore an important part of the serving configuration. Appendix D gives the capacity traces and all four comparisons.

Table 16: MI308X live width-adaptation sequence. Each row compares an $N = 9$ R3 arm with an $N = 8$ deployed-head arm inside its own four-block experiment. Errors are SEs across blocks. Rows are not a pooled factorial experiment.

Verification setting	Throughput change	CPF change	Rule passed?
Threshold, capacities not aligned	$+1.80\% \pm 0.78$	$+6.06\% \pm 0.48$	No
Fixed capacity 8 (B′)	$+6.75\% \pm 0.82$	$+4.29\% \pm 0.38$	Yes
Threshold, grid $\{2, 4, 8\}$	$+4.56\% \pm 0.35$	$+4.25\% \pm 0.38$	Yes
Stair/fill, grid $\{2, 4, 8\}$	$+7.20\% \pm 0.67$	$+4.47\% \pm 0.38$	Yes

Table 17: The fixed-capacity B′ control. Each row is a balanced block summary, not an individual request. TPS is delivered decode tokens per second.

Block	Baseline TPS	Adapted TPS	Relative TPS	Relative CPF
1	1098.8	1180.1	+7.39%	+3.18%
2	1102.9	1196.1	+8.46%	+4.64%
3	1103.4	1153.8	+4.58%	+4.46%
4	1104.0	1176.7	+6.59%	+4.86%

Each row of Table 16 compares an $N = 9$ adapted head with an $N = 8$ deployed head inside its own four-block protocol. The fixed-capacity experiment explicitly forces capacity 8. Historical grid labels in the other rows denote the observed capacity set: a later audit found that the intended environment-variable grid control was not enforced in that runtime. All 32 width logs in the grid-labeled comparisons nevertheless stay in $\{0, 2, 4, 8\}$. These are valid measured configuration contrasts, not evidence of an enforced code contract. The archived "rule passed" column uses the historical 3.2% operational threshold.

For L2, the recorded grid is $\{2, 4, 8\}$ and the observed matched-request capacity counts are $\{0 : 284, 2 : 3674, 4 : 5647, 8 : 8782\}$. W8 and W16 use the same frozen head with full verification; DEP uses C, width 10, and stair/fill. The arm orders are N–W16–W8–DEP, W16–W8–DEP–N, W8–DEP–N–W16, and DEP–N–W16–W8. Each run has one workload loop, prefill warmup, and no synchronizing round ledger; measured wall windows are 9.72–15.40 seconds. The additional contrasts W8/N and W16/N are $+19.87\% \pm 0.94\%$ and $-18.51\% \pm 0.36\%$.

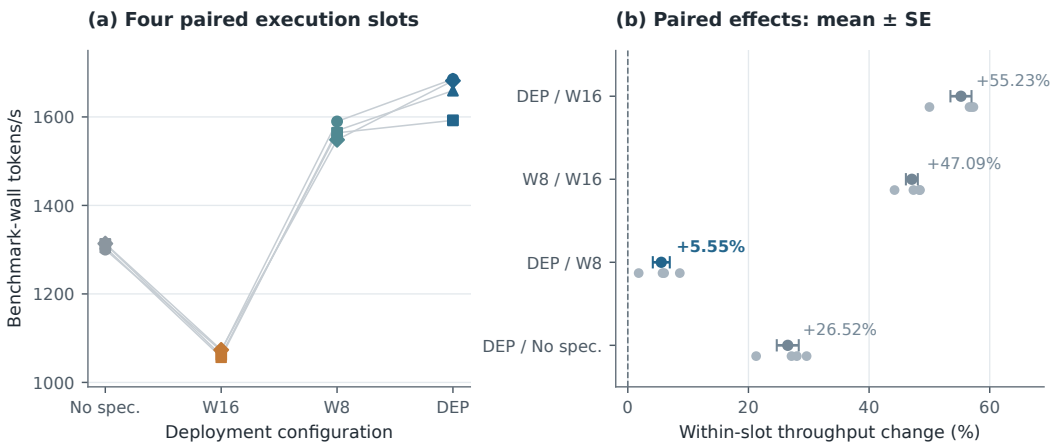

Figure 3: Joint-deployment results across four execution slots. Left: marker shapes identify slots, and lines link configurations within a slot. Right: individual slot effects and mean ± SE. The highlighted DEP/W8 contrast measures the additional configuration gain over the faster frozen static width.

D.2 Supervision-Matched Heads in Nine-Minute Serving Runs

This experiment deploys the P_m/C_m checkpoints from Appendix B.2, with no further training. Both use nine supervised positions; training widths are 16 and 10. Six deployment exports remove an unused zero-valued tier embedding and use the supported draft-model configuration. The archive records the export checks. Recorded weight fingerprints remain consistent within all six seed–head groups across calibration and evaluation.

Calibration and frozen choices. Each head is measured at query widths 8, 10, and 16 (engine draft counts 7, 9, and 15) on 72 calibration prompts, twice per width with reversed head order. This gives 36 calibration runs. The calibration uses batch ceiling 16, output cap 512, TRITON attention, model-length limit 2048, prefix caching, and prefill warmup. Within a seed, all candidates run on the same device. Selection maximizes mean decode throughput; candidates within 0.5% of the top would favor the narrower width. Every choice is unique (Table 18) and is frozen before evaluation.

Table 18: Matched-head calibration, two-run mean decode token/s. All heads independently select width 8.

Seed	Head	Width 8	Width 10	Width 16
0	P_m	**1486.1**	1374.5	1066.5
0	C_m	**1485.6**	1424.0	1080.0
1	P_m	**1518.8**	1374.9	1081.5
1	C_m	**1600.7**	1481.8	1069.8
2	P_m	**1507.9**	1382.1	1049.8
2	C_m	**1571.2**	1448.8	1097.4

Verification and stopping settings. The fixed-width branch of the benchmark passes the drafter, the DFlash method, and a fixed speculative-token count to the engine. At the selected width 8 this count is seven; the branch does not enable adaptive verification or pass a confidence threshold to the verifier. The generation call sets temperature 0 and a maximum of 512 output tokens, without custom stop strings or an override to ignore EOS. Acceptance, fallback or bonus tokens, EOS handling, and length stopping are delegated to the same engine in both arms. The trained drafter changes candidate predictions while the target-verification code is shared. These settings distinguish this comparison from the earlier adaptive-allocation experiments. As in the other batched tests, measured throughput uses actual delivered tokens; it does not assume identical floating-point generation trajectories.

Evaluation workload and timing. The frozen 900-prompt set runs for three sequential passes per engine instance (2,700 requests, 900 distinct prompts). Prefix caching and prefill warmup are disabled to avoid repeated-prefix cache benefits. A short generic kernel warmup precedes measurement. The benchmark timer spans the three workload passes, including prefill and inter-pass gaps, and excludes engine initialization. Actual benchmark windows span 545.71–560.66 s. Decode throughput uses delivered tokens after first tokens over the enclosing decode window; benchmark-wall throughput uses all delivered tokens over the full measured wall. The cache policy differs from calibration, so this evaluates the transfer of frozen choices rather than a new optimum search under the longer-window settings.

Each of three sessions includes all three seeds and two pairs per seed, with orders $P_m \to C_m$ and $C_m \to P_m$. Both arms of a pair run sequentially on one device. Seed ordering rotates across sessions, pairs are distributed over four devices, and sessions are separated by at least 30 minutes. The 36 formal results retain matching request counts and source counts within every pair, the chosen fixed width, and disabled diagnostic ledgers. Generation trajectories can differ, so each throughput uses its own delivered token count.

Table 19: All matched-head paired decode gains ($C_m/P_m - 1$, percent). Each row contains the two reversed-order pairs for one seed and session. Means are computed before rounding.

Session	Seed	P_m first	C_m first	Mean
1	0	$+2.028$	-0.040	$+0.994$
1	1	$+1.600$	$+0.909$	$+1.254$
1	2	$+1.488$	$+1.816$	$+1.652$
2	0	$+1.959$	$+2.038$	$+1.998$
2	1	$+1.668$	$+1.894$	$+1.781$
2	2	$+1.840$	$+1.850$	$+1.845$
3	0	$+1.395$	$+1.843$	$+1.619$
3	1	$+1.311$	$+1.726$	$+1.519$
3	2	$+1.623$	$+1.408$	$+1.515$

Each seed first averages its six paired ratios; the three seed means then receive equal weight. The resulting gain is 1.5753%, with SE 0.0481 percentage points across seed means. This SE describes their observed dispersion; it is not a joint uncertainty estimate over hardware and sessions. Session means are 1.3000%, 1.8748%, and 1.5509%. Full-wall throughput gives 1.5752%. For the first, second, and third workload passes separately, mean paired wall-throughput gains are 1.5509%, 1.6419%, and 1.5335%, with 18/18, 18/18, and 17/18 positive pairs. These passes reuse the same prompts and are not independent evaluations.

Order, session, and device stability. Both run orders have positive average gains, and the direction remains positive after leaving out any one session or device (Table 20). Leave-one-out estimates recompute the equally weighted seed mean from the remaining pairs. These summaries describe the sensitivity of the observed gain to the experimental grouping.

Table 20: Stability of the matched-head online gain, using the same 18 formal pairs. All entries are relative throughput gains in percent; ranges summarize the stated omissions.

Grouping or omission	Mean gain(s)
P_m first / C_m first (9 pairs each)	$+1.657/+1.494$
Sessions 1 / 2 / 3 (6 pairs each)	$+1.300/+1.875/+1.551$
Devices 0 / 1 / 2 / 3 (6 / 6 / 3 / 3 pairs)	$+1.680/+1.474/+1.611/+1.534$
Leave one session out	$[+1.425, +1.713]$
Leave one device out	$[+1.523, +1.626]$

Readiness and protocol record. An environment-only A/A precheck precedes calibration and formal evaluation. The first attempt used one workload pass and failed the duration requirement; exact delivered-token equality was also replaced by a 0.5% drift bound before treatment data. The

final protocol uses three passes with caching and prefill warmup disabled. The three final same-head runs achieve 1331.68, 1331.65, and 1313.64 decode tokens/s, giving a throughput range divided by mean of 1.361%, within the 1.5% gate, and 0.36% delivered-token drift. Their benchmark windows are 571.72, 573.80, and 579.79 s; the separate process-wall records are 625.9, 625.6, and 631.9 s.

The numerical supplement includes all three final precheck extracts and reconstructs their timing and throughput summaries alongside the 36 calibration and 36 formal records. The internal archive also retains 75 environment sidecars covering these runs, with process status and device-memory observations; all record successful completion. These sidecars are not continuous clock telemetry. The released extracts support reconstruction of the selection rule, checkpoint fingerprints, timing fields, per-pass totals, and formal paired effects.

D.3 Equal-Opportunity Width Selection and Independent Evaluation

Original-recipe summary. Separate evaluations of the original seed-0 B/C recipe also find positive mean gains after independent width selection. We evaluate B and C at query widths $\{8, 10, 16\}$ on a 72-prompt calibration workload, with four runs per candidate, and freeze each head's choice before evaluation. Both select width 8, despite their different training widths. On a separate 150-prompt, five-family workload, C improves decode throughput by $0.69\% \pm 0.32\%$ over B across 12 pairs: 10 pairs are positive, and the three session means are $+0.27/+1.21/+0.58\%$. Benchmark-wall throughput agrees ($+0.67\%$). Keeping these choices fixed on the additional 900 prompts gives a mean gain of $1.12\% \pm 0.76\%$ (SE across five workload-block means; three positive). This extension uses the original seed-0 B/C heads. Appendix D.3 reports the two evaluations separately.

Selection protocol. We use the seed-0 equal-update B/C checkpoints, MI308X, batch ceiling 16, output cap 512, TRITON attention, model-length limit 2048, prefix caching, and prefill warmup. Both heads receive the candidate set of query widths $\{8, 10, 16\}$, implemented as `fixed7`, `fixed9`, and `fixed15`. All candidates use full fixed-width verification. Calibration runs each of the six head–width combinations in four order-balanced slots on the 72-prompt primary workload (24 runs). Each head selects the largest mean decode throughput; candidates within 0.5% of the largest mean are tied, with the narrower width preferred. Both choices are unique and were frozen before evaluation.

Table 21: Equal-opportunity calibration. Values are four-run means. Bold entries are independently selected by each head using decode throughput.

Query width	B decode TPS	C decode TPS	B logged CPF	C logged CPF
8	**1537.0**	**1569.2**	3.578	3.710
10	1409.5	1411.0	3.802	3.917
16	1078.3	1049.8	4.154	4.062

Width 16 has the highest logged CPF for both heads, whereas width 8 has the highest throughput. Neither head selects its own training width. The selected configurations therefore compare B and C at the same width-8 execution setting on the independent workload.

Independent workload and execution. The evaluation collection contains 150 prompts: 30 each from unused Alpaca-derived Chinese instructions, GSM8K test items, HumanEval, MBPP, and IFEval. Exact-text disjointness checks against the calibration/primary set, distillation pool, MT-Bench set, and long-context evaluation pool passed. This collection's internal filename is `ceval150`; it is a five-family collection assembled for this experiment. Each selected head is evaluated in three sequential sessions of four pairs (24 runs). Within each pair the two heads run on different GPUs, with launches offset by 25 seconds; GPU assignment and launch order alternate. Pairs are separated by 30 seconds and sessions by 120 seconds. Diagnostic ledgers are disabled. Every evaluation run contains all 150 prompts, and checkpoint fingerprints match B/C throughout.

Table 22: Independent evaluation after selection. Entries are within-pair decode-throughput changes C/B in percent. The final row averages all 12 pairs; SE is across pairs.

Session	Pair 1	Pair 2	Pair 3	Pair 4	Mean
1	-1.68	$+2.01$	-0.76	$+1.51$	$+0.27$
2	$+1.97$	$+0.36$	$+1.51$	$+1.00$	$+1.21$
3	$+0.23$	$+0.17$	$+1.24$	$+0.68$	$+0.58$
All pairs	$+0.69\% \pm 0.32\%$ SE; 10/12 positive				

The mean benchmark-wall increment is $+0.67\%$, and mean paired logged-CPF improvement is $+1.00\%$. These measurements quantify the incremental value of the adapted head after both heads select from the same three static widths. The numerical supplement includes all 48 calibration/evaluation extracts and a script that recomputes the choices, paired effects, and session summaries. Calibration and evaluation use different workloads; their effect sizes characterize their respective protocols.

Evaluation on the additional 900 prompts. The previously selected width 8 is kept fixed for both original seed-0 B/C checkpoints. The 900 prompts are partitioned into five source-stratified workload blocks of 181, 180, 180, 180, and 179 prompts. Each block runs B–C–C–B with the same batch ceiling 16, output cap 512, and warmed serving settings. Adjacent B/C runs form two reversed-order pairs. All 20 runs retain the recorded B/C checkpoint fingerprints. These are the original serving heads, separate from the new supervision-matched training heads.

Table 23: Selected-width throughput on the 900-prompt set. Each row averages two within-block C/B decode-throughput ratios. The aggregate gives equal weight to the five blocks; SE is across their means.

Workload block	Prompts	Pair 1 C/B	Pair 2 C/B	Mean C/B
1	181	0.9978	0.9865	0.9922
2	180	0.9979	0.9943	0.9961
3	180	1.0318	1.0231	1.0274
4	180	1.0277	1.0287	1.0282
5	179	1.0101	1.0141	1.0121
Mean $\pm$ SE	900	—	—	1.0112 ± 0.0076

The mean gain is 1.12%, with three of five blocks and six of ten pairs positive. Block means span -0.78% to $+2.82\%$. This extends the earlier 150-prompt evaluation to a broader workload while preserving the calibration choices; variation across blocks remains visible in the reported estimates.

E Complete Seed Profiles and Deployment Slots

E.1 A Layout Penalty Can Be Reduced by Training

Width-10 adaptation reaches 3.7369 CPF at capacity 8, compared with 3.6656 for the frozen width-10 forward. Within the width-8 specialist, the narrow–wide gap falls from 3.19% to 0.17% (Table 5). Training at the narrower layouts improves their retained predictions and brings the width-8 specialist close to its own wider-forward performance.

E.2 The Training Advantage Reverses Across Layouts in Three Seeds

Adaptation improves the matched layout at the expense of a wider layout in all three seeds. We evaluate terminal B/C checkpoints at eight layouts. At `w10cap8`, C-minus-B pooled CPF increments are $+0.05298, +0.10426, +0.08359$. At `w17cap8`, they are $-0.03167, -0.03172, -0.02542$. All three seeds improve at full widths 4, 6, 8, 10, and 12, while full-width-16 signs are mixed (Figure 4). The sign reversal shows that the benefit of training depends on the layout used for inference.

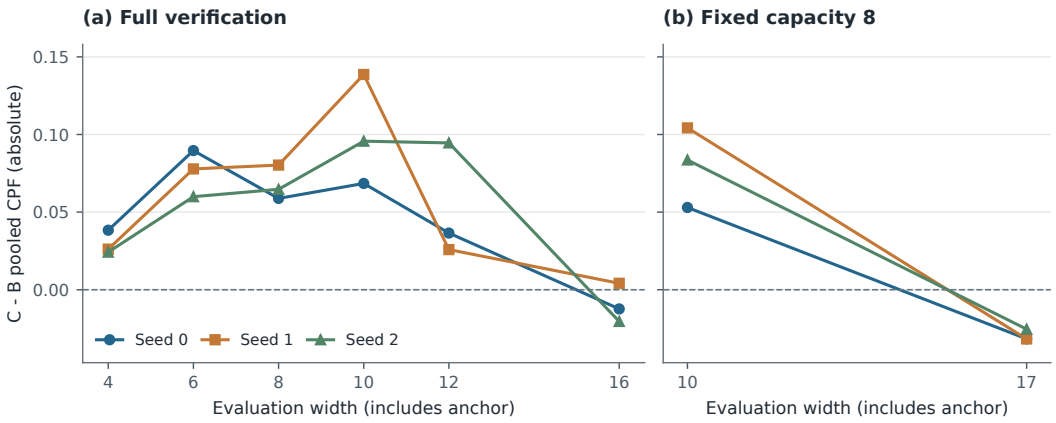

Figure 4: **Width adaptation changes which layouts the drafter predicts well at.** Paired C-minus-B pooled CPF across three training seeds and 48 prompts per layout. Left: full-verification layouts. Right: fixed capacity 8, which removes the verification-ceiling difference. The gain reverses sign between the two capacity-8 layouts in all three seeds. Lines connect measured points only.

The interaction also appears on the 900-prompt set: $(C-B)_{\texttt{w10cap8}} - (C-B)_{\texttt{w17cap8}} = +0.0738$ accepted draft tokens per block, with prompt-bootstrap interval $[0.0617, 0.0859]$. All six sources have positive intervals. Appendix B.1 reports the source breakdown and the separate 48-prompt analysis.

Public-checkpoint replication. We also compare width-specific and original-width continuation on public checkpoints. At `w10cap8`, width-10 CE improves CPF by 1.30% over equal-update width-16 CE on Qwen3-8B and by 3.78% on Qwen3-4B. Both continuation arms improve over the public initialization. Width-specific training adds to the improvement from original-width continuation (Table 6).

Mixed-width training broadens layout coverage. W-MIX lies within 0.77% of the evaluated width-specific references at widths 8, 10/capacity 8, and 16 under the same offline protocol (Table 12); the largest gap is at width 16 versus B: $1 - 4.2390/4.2718 = 0.77\%$. It also improves acceptance at untrained intermediate widths. In short-input serving, its throughput is $4.26\% \pm 0.79\%$ below C in the four-head experiment. We consequently used C for the original short-input deployment and study W-MIX separately for broader-layout and long-context transfer (Appendix C).

Table 24: P2 absolute C-minus-B pooled CPF. Each entry aggregates 48 requests; seeds use separate terminal training checkpoints. The primary gate concerns `w10cap8`.

Layout	Seed 0	Seed 1	Seed 2
w4	+0.03834	+0.02613	+0.02422
w6	+0.08964	+0.07783	+0.05993
w8	+0.05882	+0.08030	+0.06469
w10	+0.06846	+0.13866	+0.09568
w10cap8	+0.05298	+0.10426	+0.08359
w12	+0.03650	+0.02586	+0.09458
w16	-0.01236	+0.00414	-0.02044
w17cap8	-0.03167	-0.03172	-0.02542

Table 25: L2 archived benchmark-wall token/s. Within-slot ratios generate Table 15; these are execution repeats, not independent training seeds.

Slot	None	W16	W8	DEP
1	1300.12	1071.56	1589.74	1685.49
2	1312.94	1061.48	1564.09	1592.28
3	1305.19	1057.07	1569.03	1659.40
4	1313.74	1073.58	1548.32	1681.78

The L2 W8 and W16 logs report CPF 3.5849 and 4.1143, respectively, in every slot (`mixedL2s1--4W8/W16`). These are the live acceptance statistic defined in the metric conventions above. Their delivered-token/step ratios are 3.5794 and 4.0985, respectively; either accounting preserves the acceptance–throughput reversal.

Table 26: P1 A/A readiness: relative throughput of second versus first run, in percent. A card passes only if absolute mean and median absolute effect are at most 1%, and maximum absolute effect is at most 2%. These are engineering readiness criteria, not a 95% test.

Attempt	Card	Pair 1	Pair 2	Pair 3	Verdict
Initial	0	-0.127	+1.495	+3.236	Fail
Initial	1	+3.146	-0.129	-4.869	Fail
Retry	0	-0.907	+1.586	-0.992	Pass
Retry	1	+0.136	+0.353	+5.212	Fail

The retry follows stopping an ancillary GPU workload, increasing inter-run buffering, and collecting environmental samples. Other tenants remain possible, so this is not an exclusive-machine test. Of all 12 pairs, six are within 1% absolute effect and eight within 2%; their median absolute effect is 1.244%. Large deviations do not define a universal 3–5% noise floor. In the largest retry pair, total tokens change from 259,513 to 257,570, CPF from 3.5301 to 3.6114, and elapsed time from 164.42 to 155.10 s. Hardware timing alone has not been isolated as the cause.

The matched-request L2 counts exclude 25 unmatched entries per log; the larger historical count is not combined with this trace-matched summary.

F Training–Deployment Transfer Checks

Overview of workload transfer. The best deployment width also depends on the workload. We compare B/C at widths 10 and 16 on external workloads. For seed 0 on MT-Bench using the CUDA platform, the mean throughput change is $+3.03\%$ at width 10 and -0.14% at width 16. On MATH-500 the adapted head improves both widths in both slots, but the wider layout gains more: $+1.97/+2.17\%$ at width 10 versus $+3.47/+3.71\%$ at width 16. Here, training at width 10 also benefits width 16, and the larger gain is at width 16.

Across all three MT-Bench seeds, the acceptance interaction favors the matched layout. On MT-Bench using the CUDA platform, matched-width throughput changes for seeds 0/1/2 are $+3.03/-0.83/+0.45\%$; their acceptance interactions remain positive. Seed 2's positive throughput interaction arises mainly from a width-16 penalty. ShareGPT and MI308X comparisons vary across slots. Appendix F reports all cells. After adaptation, we therefore remeasure candidate widths on the intended workload.

Llama-3.1-8B has a different starting profile on both backends: its frozen head shows no detected narrow-layout penalty at fixed capacity. On the CUDA platform, both continued heads select width 8. Original-width continuation improves throughput by 8.76% over the frozen head; specialization increases logged CPF by 1.35% over that continuation but lowers throughput by 1.45% across two reversed-order slots (Appendix H).

Prompt-level interaction analysis. A retrospective reanalysis of the same checkpoints tests whether the C-minus-B effect changes with deployment layout. Unlike the pooled delivered CPF in Figure 4, this analysis first averages accepted draft tokens per block within each prompt, before final output truncation. The same-capacity interaction, $(C-B)_{\texttt{w10cap8}} - (C-B)_{\texttt{w17cap8}}$, is $+0.174$ (across-seed SE 0.011). Resampling the 48 prompt IDs jointly across the three fixed seeds gives a 95% percentile bootstrap interval $[0.076, 0.289]$ (10,000 resamples). This interval conditions on these checkpoints. Repeating the analysis with prompt-level delivered CPF, including final truncation, gives $+0.169$ with interval $[0.073, 0.281]$. Both support a layout-dependent adaptation increment; the different weighting and token accounting should not be conflated with the pooled profile.

External online cross-check. On 64 MT-Bench prompts from an external source, we evaluate the seed-0 B/C pair at draft counts 9 and 15 (training-side block widths 10 and 16). Two GPU slots reverse execution order; they are not independent temporal sessions. On the CUDA platform, C/B throughput changes are $+3.10\%/+2.97\%$ at width 10 and $-0.17\%/-0.11\%$ at width 16, giving a mean interaction of $+3.17$ percentage points. On MI308X, the corresponding interactions are $+1.20/-1.19$ points. CPF interactions have the matching direction in all four slots, while throughput evidence is hardware-dependent. Exact-text non-overlap was checked; task-family disjointness is not implied. The follow-up below reports the online training-seed variation.

Online training-seed follow-up. We subsequently evaluate seeds 1 and 2 on the same CUDA platform workload in two sequential temporal sessions per seed, reversing order and balancing devices (16 additional runs). Across seeds 0/1/2, the mean C/B throughput changes at matched width 10 are $+3.03\%/-0.83\%/+0.45\%$, whereas those at width 16 are $-0.14\%/+0.04\%/-3.43\%$. The corresponding interactions are $+3.17/-0.86/+3.88$ percentage points. Seed 2's positive interaction primarily reflects a penalty at the mismatched width, not a 3.88% speedup at the matched width. CPF interactions remain positive ($+0.0827/+0.0356/+0.0265$), supporting a consistent direction of layout-dependent acceptance adaptation in the three tested seeds. Throughput effects remain seed-dependent; these descriptive results establish neither a uniform speedup nor a noise threshold for dismissing the negative seed. The appendix retains each seed separately rather than pooling away the sign difference.

Additional external workloads. The seed-0 four-cell comparison uses 30 MATH-500 prompts and 24 ShareGPT prompts, two execution slots, and both the CUDA platform and MI308X (32 runs). The table reports each slot's C/B throughput change; it does not pool devices or workloads.

Device	Workload	Width 10 (%)	Width 16 (%)
CUDA platform	MATH-500	$1.97/2.17$	$3.47/3.71$
CUDA platform	ShareGPT	$-13.90/0.96$	$4.00/-4.57$
MI308X	MATH-500	$-1.24/0.67$	$2.87/0.53$
MI308X	ShareGPT	$-0.40/0.28$	$6.28/2.43$

On the CUDA platform MATH-500, the absolute CPF increments are 0.0782 at width 10 and 0.2411 at width 16 in both slots. The wider layout therefore gains more in both acceptance and throughput. These data support transferable adaptation benefits but do not support a universal matched-width advantage. The other cells retain their observed variability; these are two-slot, single-training-seed comparisons.

G CUDA Platform: Same-Forward Deployment Tests

The first head-only C/A experiment reports decode throughput $+6.79\% \pm 4.96\%$ and CPF $+5.62\% \pm 1.32\%$ over four blocks. A six-pair repeat uses 120 prompts and a 1,024-token output cap, versus 72 prompts and a 512-token cap previously. It yields throughput $+4.18\% \pm 2.17\%$ and CPF $+3.08\% \pm 0.55\%$, with 13–15 seconds of actual decode time per run. The workload changes preclude treating the combined estimate as a single controlled repeat.

The later 16-pair test uses 72 prompts and a 512-token output cap in one session, with alternating C/A and A/C order. Its decode-throughput effects are $[-2.29, -0.70, 9.98, 7.26, 0.37, 0.34, 12.30,$

$5.19, 7.06, -0.34, -0.06, 13.74, 5.60, 1.60, 3.51, 0.85]\%$. The mean is $4.03\% \pm 1.23\%$, median 2.56%; CPF increases $2.88\% \pm 0.33\%$. The wall-throughput effect is $4.00\% \pm 1.22\%$. Leave-one-pair-out decode means range from 3.38% to 4.45%. Wall windows span 6.75–7.66 seconds. These are run-level diagnostics, not between-session confidence intervals. The separate C/B equal-update control is retained in Table 11.

H Additional Models and Hardware Cost Regimes

Overview of the separate deployment studies.

Backend-dependent useful shapes. The MI308X draft profile falls from 8.642 to 7.213 ms between widths 9 and 10; isolated four-GEMM layer sums fall from 933.2 to 486.6 μs. The CUDA platform's corresponding layer sums are 507.5 and 509.5 μs. The two devices favor different execution choices. A Qwen3-4B width-pinned sweep also finds a slow region at $M = 144$, and a 27B/DFlash2 static comparison illustrates its practical relevance: query width 7 achieves 543.2 token/s, width 9 achieves 417.7, and no speculation achieves 418.0, although width 9 has higher CPF. The 27B training control itself is $+0.49\% \pm 0.74\%$ C/B. Cost-aware selection remains useful even where this training recipe has little measured incremental benefit (Appendix H).

Adaptation at higher concurrency. The adapted head also improves throughput at a higher concurrency ceiling. At the previously selected query width 8, C/B decode gains are $1.72\% \pm 0.52\%$ at batch 16 and $2.45\% \pm 0.90\%$ at batch 32; all eight slot comparisons improve both throughput and CPF. A separate concurrency scan places these gains in deployment context: the best tested speculative arm reaches $1.293\times$ the no-speculation throughput at batch 16, falling to $0.847\times$ at batch 32 and $0.677\times$ at batch 64. Adaptation improves the speculative path even where no speculation delivers higher throughput. Appendix H.1 reports the scan, queue-size controls, and head comparisons.

Quantized targets. With an AWQ target, the best measured adaptive arm improves over the best tested static arm by 3.0% on MI308X and 14.2% on the CUDA platform. The CUDA platform arm uses native DSpark rate-budget allocation; the MI308X arm uses threshold allocation. We remeasure costs for each hardware–kernel stack. These AWQ results compare verification allocators; the B/C training experiments use BF16 targets (Appendix H).

Long contexts. Short-context W-MIX adaptation also improves acceptance on longer inputs: CPF rises by $5.63\% \pm 1.56\%$ at 8k and by 6.01% on a 16-example real-document workload near 30k. A separate resident 30k workload shows 7.88% higher throughput in one ordered pair; 64k/96k positional-extrapolation tests have positive acceptance effects but mixed throughput. More recent 30k tests improve both attention execution and the drafter through long-context continuation, yet all tested configurations remain slower than no speculation (best observed ratio 0.497). For this workload, disabling speculation still gives the highest measured throughput. Appendix I reports the separate recipes and timing windows.

Execution-shape measurements. On MI308X, the draft-component costs at query widths 9, 10, and 17 are 8.642, 7.213, and 12.448 ms at batch 16. Isolated layer-GEMM sums at $M = 176$ and 192 are slightly below the $M = 160$ value, so width 10 is a validated useful point rather than a global cost minimum. A Qwen3-4B width-pinned experiment measures request-step-amortized costs of 1.36/2.36/1.50/1.73 ms at $M = 128/144/160/176$. These are integrated workload measurements, not isolated target-kernel timings. In its adaptive path, about 19% of steps visit padded sizes 137–152 at cap 9; a matched forced-width decomposition attributes a substantial loss to ragged execution in that region. This is evidence for measuring realized shapes, not a universal law fixing a bad row count on every model or backend.

27B/DFlash2. The static comparison uses three repeats per arm. Query widths 5/7/9 (engine draft counts 4/6/8, batch-16 row counts 80/112/144) attain median decode rates 528.89/543.19/417.72 token/s and CPF 3.3718/3.8476/4.1660. The no-speculation median is 418.02 token/s. Selecting the best tested static point improves throughput by 29.9% over no speculation. The experiment samples three widths; it neither isolates a local GEMM cusp nor tests the official block-8 setting (engine draft count 7). Its integrated time includes both drafting and verification.

Head	w8	w10cap8	w17cap8	w10	w17
A frozen	2.706	2.756	2.694	2.897	2.828
B original-width	3.010	3.044	2.956	3.272	3.247
C target-width	3.025	3.071	2.925	3.285	3.171
C−B	$+0.015$	$+0.027$	-0.031	$+0.013$	-0.075^{*}

Table 27: Llama-3.1-8B fixed-capacity A/B/C offline matrix (preregistered; per-prompt mean accepted draft tokens per block, 48 prompts, greedy, single seed). The first three columns fix verification capacity at 8 tokens and vary only draft block-context width; the last two are the full-verify secondary (capacity varies with width; descriptive only). *bootstrap CI excludes zero. Frozen-head contrasts: $\mathrm{A}_{\texttt{w17cap8}} - \mathrm{A}_{\texttt{w8}} = -0.012$ (CI $[-0.082, +0.059]$), $\mathrm{A}_{\texttt{w17cap8}} - \mathrm{A}_{\texttt{w10cap8}} = -0.062$ (CI $[-0.143, +0.005]$). Training-effect contrasts: Δ_1 (w8 vs w17cap8) $= +0.046$ (CI $[-0.033, +0.128]$), Δ_2 (w10cap8 vs w17cap8) $= +0.058$ (CI $[-0.030, +0.152]$); full-verify secondary $\Delta' = +0.088$ (CI $[+0.005, +0.181]$).

A separate four-slot training control compares width-7 specialization with equal-update original-width-8 continuation at engine draft count 6. C/B decode throughput is $+0.49\% \pm 0.74\%$ and CPF is $-0.03\% \pm 0.44\%$. The recipe was transferred and evaluated, but an incremental training benefit is not established. The data do not distinguish saturation from limitations of this particular training recipe.

Other targets. On Llama-3.1-8B, original-width continuation improves over the frozen head by $4.27\% \pm 1.29\%$ in decode throughput and $5.58\% \pm 0.74\%$ in CPF. Specialization adds $0.55\% \pm 1.59\%$ throughput and $0.12\% \pm 0.69\%$ CPF in the tested deployment. On Qwen3.5-4B, the reported B/A and C/B throughput effects are $-2.58\% \pm 0.89\%$ and $+0.50\% \pm 1.66\%$. Its recurrent-state backend uses fixed-width tests because the runtime excludes device-side verification trimming. These experiments extend implementation coverage but do not establish a general prediction of training gains from a measured calibration gap. The original Llama exclusion follows a documented kernel-JIT event and a predeclared replacement rule.

Second-family fixed-capacity probe (Llama-3.1-8B). The Llama probe tests whether the near-zero specialization gain accompanies the same narrow-layout penalty observed in Qwen3. Under the identical interaction protocol—the same 48 truncated evaluation prompts, greedy decoding, accepted draft tokens per block as the unit, verification capacity fixed at 8 block tokens, no tier conditioning—the frozen head shows no detected narrow-layout penalty across block-context widths (Table 27): $\mathrm{A}_{\texttt{w17cap8}} - \mathrm{A}_{\texttt{w8}} = -0.012$ (95% shared-prompt bootstrap CI $[-0.082, +0.059]$; single seed), excluding a loss of the Qwen3 wide-narrow magnitude ($+0.119$). The layout contrast of the training effect is correspondingly undetected: $(C-B)_{\texttt{w10cap8}} - (C-B)_{\texttt{w17cap8}} = +0.058$ (CI $[-0.030, +0.152]$) versus $+0.174$ (CI $[+0.076, +0.289]$) on Qwen3; the response intervals overlap, while the frozen-head contrast distinguishes the two tested starting profiles. Thus, the small online specialization gain accompanies a frozen head without a detected Qwen3-sized narrow-layout penalty. Continuation itself transfers: both continued heads exceed the frozen head at every layout, by $+0.26$ to $+0.42$ CPF for B and $+0.23$ to $+0.39$ for C across the five layouts. The descriptive-only full-verify secondary, where capacity varies with width, shows C worse than B at w17 (-0.075, CI $[-0.152, -0.004]$), direction-consistent with width matching; no capacity claim is made from it. The Qwen3 and Llama results motivate checking the frozen head's fixed-capacity acceptance profile before choosing a training layout.

Llama on the CUDA platform. We transfer the same frozen, original-width-continued, and width-8-specialized checkpoints to the CUDA platform. The offline probe repeats the 48-prompt, five-layout protocol above. At capacity 8, the frozen-head w17cap8-minus-w8 contrast is -0.032 (95% shared-prompt bootstrap CI $[-0.096, +0.036]$); w17cap8-minus-w10cap8 is -0.090 ($[-0.144, -0.042]$). The w10cap8-versus-w17cap8 training interaction is -0.028 ($[-0.119, +0.067]$). Both backends therefore show a frozen head without the Qwen3-sized narrow-layout penalty. The full-verification secondary interaction is $+0.084$ ($[+0.002, +0.176]$). These intervals condition on the single training seed.

Slot	No spec.	A	B	C	C/B	B/A
1	1046.11	2097.49	2305.38	2272.50	-1.43%	$+9.91\%$
2	1059.97	2140.60	2303.57	2269.80	-1.47%	$+7.61\%$

Table 28: Llama CUDA evaluation at the selected query width 8. Rates are decode token/s; effects are within-slot ratios. A is frozen, B is original-width continuation, and C is width-8 specialization. Mean C/B, B/A, and C/A effects are -1.45%, $+8.76\%$, and $+7.19\%$. Logged CPF is 3.2474/3.4316/3.4778 for A/B/C in both slots.

For serving, we first measure B and C at query widths 8, 10, and 16 (engine draft counts 7, 9, and 15), then freeze each head's choice by decode throughput. B reaches 2303.3/1976.0/1685.1 token/s and C reaches 2227.5/1961.5/1672.3; both select width 8. Two subsequent evaluation slots reverse arm order at the selected width, with frozen-head and no-speculation references (Table 28). Each run uses the same 72-prompt workload, concurrency ceiling 16, and output cap 512; calibration and evaluation are separate runs on that workload. This study comprises seven calibration and eight evaluation runs.

Continuation improves serving, while width specialization adds no throughput gain in these two slots: C has 1.35% higher logged CPF than B but 1.45% lower throughput. B reaches 2.17–2.20 times the no-speculation rate. These configuration-level speedups differ from the C/B training effect. The slots form a small within-session comparison. CUDA selection already matches C's training width, so this experiment does not call for a different hardware-specific training width.

INT4 AWQ. The same official Qwen3-8B AWQ checkpoint (group size 128) is evaluated with a BF16-distilled drafter on MI308X and the CUDA platform. Fixed-width CPF ratios relative to the BF16-target runs span 0.977–1.007. On MI308X, the adaptive threshold arm reaches 779.1 token/s versus 756.5 for the best tested static arm; on the CUDA platform, native DSpark rate-budget allocation reaches 2708.5 versus 2372.3. The latter is 32.5% above its no-speculation reference and 14.2% above its best static reference. No B/C training comparison was run with the quantized target.

The observed AWQ/BF16 no-speculation ratios differ sharply (0.419 on MI308X, 1.731 on the CUDA platform). MI308X uses a Triton AWQ path, while the CUDA platform uses a specialized weight-only quantized matrix-multiplication path. These comparisons also differ in device/session controls, including unswapped devices for the CUDA chains, so they characterize the deployed stacks rather than isolating quantization or kernel choice. CUDA platform diagnostics show mean realized width shrinking from 4.44 to 2.73 while the verification table decreases; offline cost-table swaps change the selected budget. The MI308X threshold rule leaves mean width near 6.25. Rule identity and timing boundaries are therefore necessary when interpreting cross-hardware acceptance and throughput.

Mac mini M4: selecting width without retraining. A separate llama.cpp/EAGLE3 study chooses draft ceiling 2 on calibration prompts, then evaluates six held-out prompts with four order-balanced repetitions and four arms (96 formal runs). Against native ceiling 3, the five prompts passing the predeclared exact-output gate show median decode-rate gains of 4.55%, 7.19%, 0.58%, 7.46%, and 4.68%. Their gains over ceiling 8 range from 48.37% to 67.08%. The SQL prompt fails the output/completeness gate and is not ranked. Runs generate 128 tokens with EOS ignored and exclude model loading. This single-session Mac mini result supports the broader deployment value of measuring width on a distinct stack; it is an AR-drafter configuration study, not evidence of block-diffusion training adaptation. Fill provides no consistent additional benefit.

H.1 Concurrency, Queue Size, and Head Adaptation

Protocol. The MI308X concurrency study fixes the target, TRITON attention, output cap 512, model-length limit 2048, prefix caching, and prefill warmup. The concurrency ceiling is the engine's maximum number of active sequences. A 20-run scan crosses ceilings $\{16, 24, 32, 48, 64\}$ with no speculation, `fixed4`, `fixed8`, and the existing `tierspec` configuration, using the frozen domain head for speculative arms. The command requests up to 30 prompts per family; the available set supplies **161 actual prompts**. Eight additional runs use 48 prompts at ceilings 16 and 64 to characterize queue-size sensitivity. Each scan cell is a single run. Runs use four reserved devices; the

archived post-run memory checks flag no contamination. An earlier resource-contaminated attempt is excluded from these results.

Table 29: Concurrency scan with 161 prompts. No-speculation rates are decode tokens/s; speculative entries are ratios to the same-row no-speculation rate. Bold marks the best tested speculative arm.

Concurrency ceiling	No speculation	`fixed4`	`fixed8`	`tierspec`
16	1344.6	**1.293**	0.786	0.873
24	1743.1	**0.920**	0.861	0.724
32	2129.9	**0.847**	0.644	0.699
48	2399.6	0.669	**0.727**	0.644
64	2915.8	**0.677**	0.609	0.523

The best tested speculative arm is faster than no speculation at ceiling 16 and slower at the remaining sampled ceilings. Increasing the ceiling changes both batch execution and the relative efficiency of the no-speculation reference. The fixed4 ratio at ceiling 16 remains positive with either queue size (Table 30). At ceiling 64, a 48-prompt queue cannot fill all 64 sequence slots, while the 161-prompt workload provides enough queued requests to do so.

Table 30: Queue-size controls. Each entry is speculative/no-speculation decode throughput within the same workload and concurrency ceiling.

Ceiling	Arm	48 prompts	161 prompts
16	`fixed4`	1.268	1.293
16	`fixed8`	0.934	0.786
16	`tierspec`	0.955	0.873
64	`fixed4`	0.806	0.677
64	`fixed8`	0.707	0.609
64	`tierspec`	0.622	0.523

Same-width head comparisons. A separate 16-run experiment evaluates the seed-0 B/C heads at query width 8 (`fixed7`), selected in Appendix D.3. Both heads process the same 161-prompt workload at ceilings 16 and 32, with four execution slots per head/ceiling. Jobs are queued across four devices; each slot pairs B/C at the same ceiling. Checkpoint fingerprints match the intended heads in all 16 records. The following effects are computed within slot before averaging; SE is across the four slots.

Table 31: C/B adaptation effects at fixed query width 8. Slot entries and means are decode-throughput changes in percent. CPF improvements are positive in all eight comparisons.

Ceiling	Slot 1	Slot 2	Slot 3	Slot 4	Mean $\pm$ SE
16	$+3.22$	$+1.14$	$+1.64$	$+0.89$	$+1.72\% \pm 0.52\%$
32	$+2.65$	$+4.69$	$+0.33$	$+2.15$	$+2.45\% \pm 0.90\%$

Benchmark-wall gains are $+1.71\%$ and $+2.41\%$, respectively. Mean B/C decode rates are 1447.2/1472.2 tokens/s at ceiling 16 and 1540.4/1578.0 at ceiling 32. Thus the adapted head improves the same-width speculative path at both tested concurrency levels. The separate scan evaluates whether speculation should be enabled: at ceiling 32 its no-speculation reference reaches 2129.9 tokens/s, compared with 1804.1 for its best tested speculative arm (fixed4). All 44 replacement-study records and numerical reconstructions are included in the supplement.

I LONG-CONTEXT ACCEPTANCE AND NET ACCELERATION

Table 32: Long-context live evidence, W-MIX versus the deployed head under matched within-row configurations. The statistical design differs by row. "Public" denotes public task sources with natural-text filler, not a long-context task-accuracy benchmark.

Workload	Repeats / prompts per run	CPF change	Throughput change
8k mixed	3 triples / 48	$+5.63\% \pm 1.56$	$+4.89\% \pm 3.44$
30k mixed, cache-evicting	1 pair / 120	$+11.38\%$	$+1.64\%$
30k real documents	1 pair / 16	$+6.01\%$	$+2.00\%$
30k public, resident	1 ordered pair / 24	$+13.13\%$	$+7.88\%$
64k public, YaRN	2 opposite-order pairs / 12	$+8.6\%$	Inconclusive
96k public, YaRN	2 opposite-order pairs / 6	$+9.9\%$	Inconclusive

Table 33: Full 64k/96k pair effects. The second pair at each depth reverses head order. Changes in summed request steps are not changes in engine batch-round time.

Context	Pair	CPF	Request steps	TPOT median	Throughput
64k	1	$+7.9\%$	-4.8%	-4.6%	-6.6%
64k	2	$+9.3\%$	-4.6%	-12.28%	$+2.5\%$
96k	1	$+9.5\%$	-6.6%	-13.54%	-0.9%
96k	2	$+10.4\%$	-11.8%	-14.12%	$+19.2\%$

These tests use W-MIX, not the main short-input specialist C. Offline natural-text-extended 8k prompts improve CPF from 2.3828 to 2.5304 at width 4, 2.9769 to 3.1466 at width 8, and 3.0170 to 3.1819 at `w10cap8`. On 16 real-document examples near 30k, width-8 CPF improves from 1.4090 to 1.4689. Different workloads and protocols prevent interpreting these values as a controlled context-length decay curve.

The 120-prompt 30k workload has a working set around 3.6 million tokens, exceeding the approximate one-million-token cache capacity. A separate 24-prompt resident public-task workload reports 77.65 versus 83.77 wall token/s (+7.88%) and CPF 1.8415 versus 2.0833. Because workload composition also changes, this is not a causal cache-residency comparison. The real-document test reports +2.00% throughput in one pair. At 64k/96k, target and drafter configurations both use YaRN factors 1.6/2.4 (Peng et al., 2023); neither head is trained under YaRN. These runs test symmetric positional extrapolation, not long-document answer accuracy or learned context-window extension.

A separate 8k comparison finds the adapted arm 26.35% slower than disabling speculation. At batch 32 in the longer-output protocol, its deficit is 24.16%. A batch-32 fixed-capacity-4 comparison improves CPF by 3.72% but changes throughput by $-4.99\% \pm 1.55\%$. This distinguishes improving an existing speculative path from achieving net acceleration over ordinary decoding.

Updated 30k cost and training interventions. A resident LongBench-derived workload with 12 prompts compares TRITON and FLASH attention on the CUDA platform. No-speculation throughput rises from 169.66 to 188.67 token/s. Speculative engine draft counts 4/8/16 rise from 57.52/50.49/51.92 to 88.06/77.85/73.56 token/s. Despite these substantial backend improvements, the best of those speculative arms is only 0.467 times its no-speculation reference.

A separate long-context continuation uses filtered Chinese Wikipedia documents (median about 19.5k input tokens), target-generated continuations, and a mixture with short training data. Its held-out leading-prefix acceptance probe improves, and its width-16-trained head increases live CPF at engine draft count 16 from 1.759 to 2.011 on the CUDA platform and from 1.740 to 2.052 on MI308X. Note that engine draft count 16 executes query width 17. The two CUDA throughput changes versus the base head are $-2.95/+19.02\%$; MI308X changes are $+9.90/+19.21\%$. All

tested arms remain slower than no speculation, with the highest individual ratio 0.497. The offline probe, training documents, and online English long-document workload differ in metric and domain, so their ratios do not measure a causal transfer fraction. These are tested operating limits, not an impossibility result for 30k speculation.

Output diagnostics. The original 30k small-output audit records exact agreement in 8/8 repeated base outputs, 6/8 repeated mixed-width outputs, and 6/8 cross-head outputs. Later complete-output short-workload diagnostics include two differing sequences out of 12 despite identical aggregate token counts (2678 in both runs); one fixed-width repeat differs on 4/12 sequences. Near-tie logits, including 0.25-nat margins, occur in the diagnostics, but neither cache effects nor confidence timing was isolated as the cause. We therefore retain actual delivered-token counts and report performance without asserting bitwise output equivalence or task-quality gains.

J Sustained-Window Readiness Protocol

The separate P1 protocol planned eight balanced blocks (32 formal runs) comparing frozen and adapted heads at aligned deployment settings, conditional on an A/A readiness check. A pilot selected 16 workload loops after eight loops produced only 79.30/81.78 s and sixteen produced 155.98/158.30 s on the two cards. Each run uses a full untimed workload warmup followed by repeated workload loops in one engine. Per card, three same-configuration A/A pairs must have absolute mean effect at most 1%, median absolute effect at most 1%, and maximum absolute effect at most 2%.

The first attempt failed on both cards; one allowed infrastructure adjustment and retry passed card 0 but failed card 1 (maximum absolute effect 5.21%). Execution stopped as specified, with zero formal treatment runs. Appendix E gives all A/A pairs. The outcome is *uncompleted sustained-throughput confirmation*, not a negative estimate of the method's treatment effect. Same-head output counts and CPF also changed, so the observed variation is not identified as purely hardware timing noise. L2 is a different short-window experiment, not a replacement pass for P1. The later matched-head experiment in Appendix D.2 completes a separate nine-minute serving protocol; it does not retroactively pass this earlier gate.

K Native Allocator Comparison

Earlier CUDA platform experiments compare native DSpark (D), frozen stair allocation (T), and stair plus fill (F), before the width-adaptation intervention studied here. A 36-run, two-session comparison gives F/D $+1.67\% \pm 0.98\%$, F/T $+0.80\% \pm 1.07\%$, and T/D $+0.89\% \pm 0.48\%$. An independent 48-run follow-up, 24 pairs across four sessions, gives F/D $+0.29\% \pm 0.59\%$; session means are $+1.03, -0.06, +1.90, -1.71\%$. These results establish neither superiority nor equivalence.

Read-only action checks validate native budget selection and exact T/F realization. A subsequent paired-snapshot diagnostic confirms that native realization is the top-k allocation on its newer confidence snapshot in 539/539 states. The snapshot difference therefore does not establish a native execution error. Both the original and repeated speed tests disable diagnostic hooks. The follow-up also records output trajectories: 64.2% of 2,880 aligned request pairs are token-identical, with differences concentrated in long free-text tasks. These are allocator and execution observations, distinct from the C/B training intervention.

The original external cost-table replacement experiment required correcting our own measurement boundary, coverage beyond graph capture, and profiling context length. Corrected native-table comparisons did not establish an additional speed gain. Those implementation diagnoses motivate compatible cost measurements but do not imply a defect in the upstream native allocator.

L Exploratory Cost-Proxy Replay and External Acceptance Artifacts

L1 explores whether acceptance-only and cost-conditioned choices agree. Four frozen full-verification measurements at widths 8, 10, 12 and 16 yield TPS

$[1504.42, 1406.56, 1229.10, 1049.43]$ and CPF $[3.5536, 3.7875, 3.8659, 4.1143]$. The replay defines an empirical cost proxy as benchmark wall time divided by `target_rounds`. In the benchmark implementation this field sums *request* verification steps, not GPU batch rounds. The resulting costs $[2.35907, 2.68917, 3.13463, 3.90542]$ ms are per-request-step amortized wall costs under that workload.

For each of 48 prompts, acceptance selection maximizes its observed CPF over the four widths; cost selection maximizes CPF divided by the proxy. A selected policy aggregates total tokens over the sum of selected block counts times proxy costs. For A/B/C/D heads, respectively, the cost-to-acceptance replay gains are 24.485%, 24.200%, 23.765%, and 22.149%; choices agree for 13/48, 13/48, 15/48, and 13/48 prompts. Gains over the best static proxy choice are only 1.729%, 1.939%, 1.588%, and 1.742%. All values reproduce the archived micro calculation. They assume transferable additive costs across heads and prompts, despite batch occupancy and varying output work, and are not measured dynamic-serving gains. L2 independently tests static widths and DEP; it does not validate the full replay model.

BlockPilot acceptance artifacts. The archive records upstream commit `68c2fa2d76a020f863da6c58755cf5f0f807e3b6` and both released and locally reproduced acceptance tables for GSM8K and MATH500, 32 samples each, at widths 13–19. The reproduction wrapper disables FlashAttention imports to use the benchmark's SDPA fallback. The tables support an acceptance high-value region near widths 16–17, not identical exact peaks: the released GSM8K/MATH500 maxima are at 17/16, while the local maxima are at 16/17. Local MATH500 means are 8.11138 at width 16 and 8.31074 at width 17. Exact prompt identities and the external source tree are not fully archived, limiting independent attribution of sample differences.

External replay borrows the same empirical costs: widths 13–15 are linearly interpolated between 12 and 16; widths 17–19 are *extrapolated*. Published versus local cost-conditioned replay gains over fixed width 16 are 9.659%/9.292% for GSM8K and 8.131%/7.810% for MATH500. These arrays contain one mean acceptance scalar per sample rather than complete token/block counts, and do not define an end-to-end serving comparison. The released `1.json` width choices match acceptance argmax on 31/32 and 29/32 samples, but their generation provenance is not established by the stored arrays alone; we do not treat that agreement as a full predictor reproduction. The external artifacts contextualize the acceptance objective, while the paper's speed claims use the actual within-protocol serving comparisons.

M Reproducibility and Statistical Conventions

Cost profiles record checkpoint, backend, batch, context, capture settings, and timing boundaries. Paired arms share prompts, generation limits, cache policy, and runtime tree; capacity traces record realized allocations. For delivered tokens Q and measured time T, throughput obeys

$$\log \frac{\mathrm{TPS}_2}{\mathrm{TPS}_1} = \log \frac{Q_2}{Q_1} - \log \frac{T_2}{T_1}. \tag{4}$$

Request steps differ from GPU batch forwards. Prompt analyses resample shared IDs jointly across fixed seeds; timing estimates use their stated repetition units. We retain archived engineering gates and report effect sizes and variability.

The numerical supplement contains selected numeric fields from 350 serving runs and thirty-eight offline probes, excluding prompts, generated text, and machine paths. Its manifest hashes released extracts; the internal source map links them to raw logs and scripts. Coverage includes the 900-prompt source analyses, matched-supervision intervals, nine training records, the twenty-run original-head extension, 72 matched-head calibration/formal runs, three A/A prechecks, and the completed three-width acceptance profiles. Remaining appendix studies and full online reproduction are outside its scope. Manuscript sources are separate.

The script `reproduce_current.py` reconstructs the earlier training, CUDA, external-workload, and 27B comparisons, recording SHA-256 hashes for 132 source runs. Live reproduction requires the patched vLLM base, dependencies, patch order, checkpoints, and launch settings. Throughput runs disable synchronizing diagnostic ledgers; prefix fingerprints remain distinct from full checkpoint hashes.