TRACEDRAFT: ONLINE DRAFT ADAPTATION DURING SPECULATIVE DECODING

Anonymous authors
Paper under double-blind review

ABSTRACT

Speculative decoding, which verifies draft tokens in a single target-model pass, has been widely adopted as an effective method for accelerating language model inference. Online distillation has been used to adapt speculative drafts to the requests encountered during deployment. However, methods that train on completed requests or request batches defer the benefit of their feedback to subsequent requests. Consequently, exploiting token-level feedback before the current response ends, which enables immediate adaptation from ongoing inference without first generating complete responses for training, remains a challenge. As real request streams often span multiple domains, retaining useful adaptation across domain changes is also worth addressing. To tackle these challenges, we propose TraceDraft, a lightweight framework for online draft adaptation within responses. TraceDraft uses target predictions on the prompt and verified continuation as supervision, updating only the draft's output normalization and prediction head. Two training modes are available to deal with different situations: Reset mode confines adaptation to each conversation, while Persistent mode retains updates across requests with a built-in anchor toward the initial weights. Updates of TraceDraft remain separate from outer training, enabling combination with other online learners. We evaluate eight workloads, including AIME2024/2025 and MT-Bench. Experimental results show gains in accepted length and end-to-end throughput on all eight workloads, with long-output gains on DeepSeek-R1-Distill-Llama-8B averaging 10.84% and 9.16%, respectively, over the fixed draft. Combining TraceDraft with online learners also adds up to 10.03% throughput over their respective baselines. In cross-domain experiments, Persistent improves throughput by up to 21.11% over the full stream and up to 12.10% after the domain switch relative to the fixed draft. These results suggest that TraceDraft provides a practical framework for learning from incoming requests to improve speculative decoding.

1 INTRODUCTION

Autoregressive language models generate tokens sequentially, making long responses costly to serve. Therefore, speculative decoding has emerged as a significant method to accelerate language model inference, which reduces target-model evaluations by verifying a draft continuation in parallel (Leviathan et al., 2023; Chen et al., 2023a). Widely used speculative decoding works include EAGLE-3, which improves draft quality through training-time rollout (Li et al., 2025) and FR-Spec, which reduces vocabulary projection cost (Zhao et al., 2025). Both draft quality and proposal cost determine the resulting speedup.

In recent years, online distillation has been applied to speculative decoding to accelerate inference by adapting drafts to incoming requests. OSD adapts drafts to observed queries through distillation (Liu et al., 2024). Test-Time Speculation (TTS), an arXiv preprint, updates the draft during a response to counter declining acceptance on long generations (Kumar et al., 2026). Our method develops this setting through committed-prefix supervision, cache-preserving updates, and controlled state reuse. OnlineSPEC relates acceleration to dynamic regret and develops optimistic and ensemble learners (Qian et al., 2026). All these works demonstrate the value of adaptation at different timescales.

However, effective adaptation depends on when an update pays off and how long it remains useful. In the released Ens-EAGLE-3 pipeline by OnlineSPEC (Qian et al., 2026), an update becomes avail-

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.

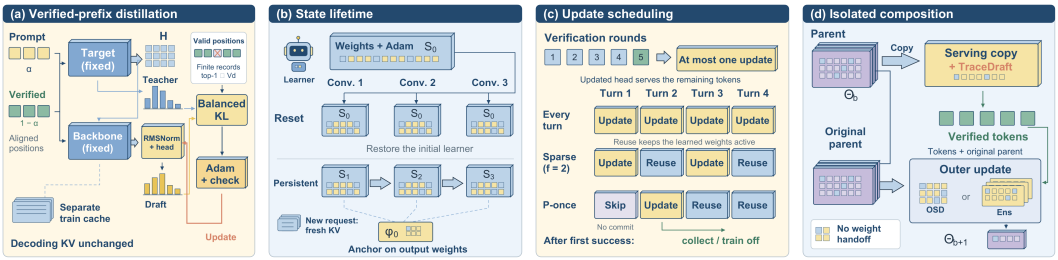

Figure 1: Core mechanisms of the TraceDraft framework. (a) Verified-prefix distillation aligns the prompt and committed continuation with target supervision, keeps the target and EAGLE-3 backbone fixed, and updates only the output normalization and prediction head without changing the decoding KV cache. (b) Reset restores the initial learner between conversations, whereas Persistent carries weights and optimizer state across requests while anchoring the output weights to their initialization; request KV caches remain local. (c) The scheduler permits at most one update after five verification rounds on a selected turn and supports every-turn, sparse, and P-once schedules. P-once stops supervision collection and training after the first successful update. (d) When TraceDraft is combined with OSD or an ensemble learner, serving adaptation uses a temporary copy, while the outer learner receives the generated response and updates from the original parent, keeping the serving weights outside the outer parameter trajectory.

able only after its source requests finish. Learning within a response can exploit feedback sooner and cover the training cost. Real request streams also have changing domains, making state retention consequential: useful updates can save repeated training, but carrying unsuitable changes forward can reduce draft–target agreement. Timely learning and controlled reuse must therefore be considered together.

To address these challenges, we introduce **TraceDraft**, a framework for adapting speculative drafts within responses and across requests. Our core intuition is that token prediction can adapt while feature computation remains fixed. Built on EAGLE-3 (Li et al., 2025), it learns from the prompt and verified continuation by updating only the output normalization and prediction head. This preserves decoding caches and lets the updated head guide the remaining response. TraceDraft supports Persistent mode, which carries adaptation across requests and includes an anchor toward the initial weights. Retained calibration can benefit related prediction patterns in a new domain, while Reset provides a fresh starting point when earlier adjustments interfere. TraceDraft also has isolated serving weights, which enable combination with outer online learners.

Specifically, our key contributions are as follows.

- We formulate **the joint problem of update timing and reuse**: benefiting the response that supplies supervision while retaining useful adaptation across changing request domains. This connects immediate decoding gains to the needs of a continuing request stream.
- We design **TraceDraft**, combining cache-preserving distillation with configurable state retention and update schedules. Its separation of serving updates from outer training makes these mechanisms complementary to existing online learners.
- We provide **theoretical analysis and extensive experiments**. We establish cache invariance, conditions for recovering update costs, and isolation of outer training. We evaluate eight workloads on DeepSeek-R1-Distill-Llama-8B and Vicuna-13B-v1.3, averaging results over at least three runs. TraceDraft improves accepted length and end-to-end throughput on every workload for both targets, with mean long-output gains on DeepSeek of **10.84%** and **9.16%** over fixed EAGLE-3. Combinations with online learners add up to **10.03%** throughput over their respective baselines; freezing preserves gains without continued training, and domain-switch experiments show when retaining state is preferable to resetting it.

Our framework provides a pragmatic path towards building a more efficient speculative decoding methodology.

2 THE TRACEDRAFT FRAMEWORK

The TraceDraft framework is designed to further accelerate speculative decoding through online distillation within requests. This section presents its architecture and key mechanisms, together with theoretical results on cache preservation, update costs, and composition with other learners.

TraceDraft updates the draft's output layers from the prompt and verified prefix, then controls how long the learned state remains in use. We first describe how an update is constructed, applied, and retained. We then introduce its scheduling controls and integration with outer online learners. Figure 1 summarizes the four mechanisms that connect within-response supervision, state retention, update scheduling, and isolated composition. Algorithm 1 gives the procedure.

Problem formulation. Each conversation $\mathcal{C}_i$ contains one or more generation turns. With the target and greedy verification fixed, we seek net decoding savings by learning only from the prompt and continuation committed so far.

2.1 VERIFIED-PREFIX DISTILLATION

To update predictions without rebuilding decoding caches, we freeze the EAGLE-3 backbone G_ψ and target parameters θ, and train only $\phi = (\gamma, W)$: the output RMS normalization scale and prediction head. Unlike TTS, which distills target predictions on a draft-generated canvas (Kumar et al., 2026), TraceDraft uses the prompt and target-committed continuation. The following four design choices specify how supervision is aligned, balanced, applied, and retained.

(1) Prefix alignment. For committed tokens $x_0, \ldots, x_t$, target features H_j and logits z_j follow consumption of x_j, so z_j predicts x_{j+1}. EAGLE-3's shifted feature–token inputs yield the aligned draft prediction

$$f_j = G_\psi(H_{0:j-1}, x_{1:j})_{\text{last}}, \qquad d_{\phi,j} = W\,\text{RMSNorm}_\gamma(f_j), \quad j \geq 1. \tag{1}$$

We reuse committed target states and recompute draft features with a separate training cache, excluding rejected branches. Feature extraction and teacher-logit projection count toward adaptation cost.

(2) Source-balanced supervision. We retain finite records whose full-target argmax lies in the draft vocabulary $\mathcal{V}_d$ and distill at temperature T (Hinton et al., 2015):

$$p_j^T = \text{softmax}(z_j|_{\mathcal{V}_d}/T), \qquad q_{\phi,j}^T = \text{softmax}(d_{\phi,j}/T), \qquad \ell_j(\phi) = T^2 D_{\text{KL}}(p_j^T \| q_{\phi,j}^T). \tag{2}$$

Verification still uses the full target vocabulary. To prevent long prompts from dominating, we normalize sampled prompt and response positions $\mathcal{I}_p, \mathcal{I}_r$ separately:

$$\mathcal{L}(\phi) = \sum_{g \in \{p,r\}} \frac{\bar{w}_g}{|\mathcal{I}_g|} \sum_{j \in \mathcal{I}_g} \ell_j(\phi), \qquad \bar{w}_g = \frac{w_g \mathbf{1}[|\mathcal{I}_g| > 0]}{\sum_h w_h \mathbf{1}[|\mathcal{I}_h| > 0]}, \quad (w_p, w_r) = (\alpha, 1-\alpha). \tag{3}$$

Empty groups are omitted and weights renormalized. Adam (Kingma & Ba, 2015) updates an FP32 master copy; checked candidates are published to the inference head, while nonfinite values or excessive relative parameter steps trigger rollback.

(3) Cache preservation. Restricting updates to output layers preserves the committed-prefix caches, as the following proposition establishes.

Proposition 1 (Cache invariance)**.** *For a fixed committed prefix, target parameters θ, and draft backbone ψ, changing only ϕ leaves the cached target and draft keys and values unchanged.*

Proof. Both caches depend on the prefix and frozen parameters (θ, ψ), upstream of ϕ. Subsequent proposals therefore reuse these tensors with the updated output scores. □

Gradients, optimizer moments, and rollback are also confined to the output layers, avoiding backbone backward computation.

(4) State lifetime. Reset limits the accumulation of request-specific changes; Persistent retains calibration for streams that change domain. Let S_i contain output weights, the FP32 master copy,

Algorithm 1 TraceDraft with a frozen EAGLE-3 backbone

Require: Frozen target θ, backbone ψ, initial learner S_0, mode, interval f
$S \leftarrow S_0$; $q \leftarrow 0$; frozen $\leftarrow$ false
for conversation $\mathcal{C}_i$ **do**
  If Reset: $S \leftarrow S_0$; restore S
  for generation turn in $\mathcal{C}_i$ **do**
    $q \leftarrow q + 1$; prefill, then draft and verify a warmup prefix
    if selected(q, f), not frozen, and generation remains **then**
      Build aligned supervision $\mathcal{B}$; skip adaptation if unusable
      $S' \leftarrow \mathrm{AdamCandidate}(S, \nabla_\phi \mathcal{J}(\mathcal{B}))$
      if S' passes checks **then**
        $S \leftarrow S'$; publish inference weights; if P-once: frozen $\leftarrow$ true
      else
        Restore S
    Complete the answer with the current head; no further update in this turn

optimizer moments, and bookkeeping. Their initialization differs:

$$S_i^{\text{start}} = \begin{cases} S_0, & \text{Reset}, \\ S_{i-1}^{\text{end}}, & \text{Persistent}. \end{cases} \tag{4}$$

Reset shares state within a conversation and restores S_0 between conversations. Persistent carries state across requests, with request-specific decoding caches. Its built-in anchor regularizes displacement from the initial weights (Li et al., 2018). Whereas Reset minimizes $\mathcal{J} = \mathcal{L}$, Persistent uses

$$\mathcal{J}(\phi) = \mathcal{L}(\phi) + \lambda \Omega(\phi, \phi_0), \qquad \Omega(\phi, \phi_0) = \sum_{u \in \{\gamma, W\}} \frac{\|u - u_0\|_F^2}{\max(\|u_0\|_F^2, \epsilon)}, \tag{5}$$

where $\epsilon > 0$ stabilizes normalization; domain-switch experiments use $\lambda = 0.01$. Retained calibration can help domains with shared predictive structure. The following condition explains why conflicting adjustments may instead favor Reset.

Remark 1 (Changing requests). For a β-smooth future distillation loss $\mathcal{L}_{\text{next}}$ and committed step $d = \phi^+ - \phi$,

$$\mathcal{L}_{\text{next}}(\phi + d) \le \mathcal{L}_{\text{next}}(\phi) + \langle \nabla \mathcal{L}_{\text{next}}(\phi), d \rangle + \frac{\beta}{2} \|d\|^2. \tag{6}$$

Thus $\langle \nabla \mathcal{L}_{\text{next}}(\phi), d \rangle < -\frac{\beta}{2}\|d\|^2$ suffices to reduce future loss. This is a condition on distillation loss; acceptance is measured separately.

Update-cost analysis. These mechanisms improve service time only when decoding savings repay adaptation. Let $U > 0$ include initialization or Reset restoration, supervision, training, checks, and publication; $C \ge 0$ is common work before the update. For n remaining returned tokens, assume stationary mean decoding costs c_0, c_1 per token before and after adaptation, excluding U.

Theorem 1 (Amortization over a continuation)**.** *If* $0 < c_1 < c_0$*, then*

$$\mathcal{S}(n) = \frac{C + nc_0}{C + U + nc_1} > 1 \quad \Longleftrightarrow \quad n > \frac{U}{c_0 - c_1}, \qquad \frac{\partial \mathcal{S}}{\partial n} = \frac{C(c_0 - c_1) + c_0 U}{(C + U + nc_1)^2} > 0. \tag{7}$$

Moreover, $\lim_{n \to \infty} \mathcal{S}(n) = c_0/c_1$.

Proof. The time saved is $n(c_0 - c_1) - U$. Differentiating the ratio gives Equation (7); dividing numerator and denominator by n gives its limit. □

Longer continuations therefore help amortize useful early updates. With variable per-token savings δ_t, the exact condition is $\sum_{t=1}^{n} \delta_t > U$.

2.2 Update Scheduling

State retention and update scheduling govern different decisions: Reset and Persistent determine how learner state crosses conversation boundaries, whereas freezing and sparse schedules determine when optimization occurs. P-once and the sparse variants evaluated here retain Persistent state. We introduce the following two controls to reduce recurring training costs.

(1) Turn selection. For interval f, generation turns satisfying $(q-1) \bmod f = 0$ permit at most one update after an initial verified prefix. The head remains active between updates. Over Q turns, this allows at most $\lceil Q/f \rceil$ updates; fewer updates reduce training opportunities but may also reduce decoding savings.

(2) First-successful-update freezing. P-once retains the first successfully committed weights, then disables training and supervision collection. Skipped or rolled-back attempts do not trigger freezing, which can occur within a turn rather than at a conversation boundary.

Scheduling-cost analysis. After a common first update in turn $q_\star$, let $\delta_q = C_q^{\text{freeze}} - C_q^{\text{cont}}$ denote later decoding-time savings from continued adaptation. Let u_q include its additional collection, training, rejected attempts, and state-handling costs. Then

$$\mathcal{T}_{\text{cont}} - \mathcal{T}_{\text{freeze}} = \sum_{q > q_\star} (u_q - \delta_q). \tag{8}$$

Continued learning therefore pays off when additional savings exceed recurring costs. Section 4.2 tests early calibration and later maintenance through freezing and sparse updates.

2.3 Composition with Online Learners

OSD (Liu et al., 2024) and the released Ens-EAGLE-3 pipeline (Qian et al., 2026) train on completed request blocks, whereas TraceDraft updates within responses. We combine these timescales by separating serving updates from outer training, then account for their joint cost.

Separate parameter histories. Let Θ_b include the outer checkpoint, optimizer, and ensemble state for block b. Each composed Reset conversation uses a serving copy of that checkpoint. Generated tokens form $\mathcal{D}_b$, while outer training retains its own parent:

$$\phi_{b,i}^{\text{serve}} = \text{TraceDraft}(\Theta_b; \mathcal{C}_i), \qquad \Theta_{b+1} = \text{OuterUpdate}(\Theta_b, \mathcal{D}_b). \tag{9}$$

Ensemble training and merging follow the selected OnlineSPEC parent rule and meta-learner. Backbone replacement reconstructs the serving learner and caches. This separation preserves the outer trajectory under the following conditions.

Proposition 2 (Isolation of the outer trajectory)**.** *Suppose paired runs share the initial outer state, prompts, exact greedy target verification and tie rule, and deterministic outer training and meta-updates based only on generated tokens, outer state, and matched seeds. If temporary TraceDraft parameters never enter the outer checkpoint, both runs have identical outer trajectories.*

Proof. Exact greedy verification gives identical tokens and $\mathcal{D}_b$. Identical outer states and deterministic updates then yield identical next states; induction completes the argument. □

Section 4.1 checks output tokens and outer checkpoints in floating-point execution.

Combined-cost analysis. Identical trajectories need not imply identical elapsed times. For block b, let U_b include all TraceDraft overhead, including Reset restoration; let $\mathcal{D}_b^{\text{time}}$ be decoding-time savings and ΔO_b the change in outer training, validation, checkpoint handling, and merging time. Then

$$\mathcal{T}_{\text{combined}} - \mathcal{T}_{\text{outer}} = \sum_b \big(U_b - \mathcal{D}_b^{\text{time}} + \Delta O_b\big). \tag{10}$$

The combination is faster when decoding savings exceed the added costs. Equal outer costs cancel in this difference, while both measured throughput denominators retain them.

Table 1: Accepted length (mean, with standard deviation where available) and end-to-end speedup across eight workloads.

Datasets	**MT-Bench**		**PubMed**		**PubMedQA**		**TheoremQA**	
Method	avg len / speedup		avg len / speedup		avg len / speedup		avg len / speedup	
Target model: DeepSeek-R1-Distill-Llama-8B								
EAGLE-3	4.29±0.02	1.00×	3.41±0.01	1.00×	3.72±0.01	1.00×	4.69±0.02	1.00×
TraceDraft	**4.49±0.04**	**1.02×**	**3.55±0.02**	**1.03×**	**3.88±0.02**	**1.02×**	**4.88±0.03**	**1.02×**
OSD-EAGLE-3	4.73±0.03	1.11×	3.62±0.02	1.05×	3.91±0.01	1.05×	5.05±0.03	1.15×
OSD-TraceDraft	**4.87±0.02**	**1.13×**	**3.69±0.01**	**1.09×**	**3.92±0.02**	**1.06×**	**5.18±0.04**	**1.18×**
Ens-EAGLE-3	4.83±0.03	1.13×	3.73±0.02	1.09×	3.96±0.02	1.08×	5.11±0.01	1.17×
Ens-TraceDraft	**4.89±0.02**	**1.17×**	**3.79±0.01**	**1.11×**	**4.05±0.02**	**1.09×**	**5.20±0.01**	**1.21×**
Datasets	**AIME2024**		**AIME2025**		**LiveCodeBench**		**LongBench-Write**	
Method	avg len / speedup		avg len / speedup		avg len / speedup		avg len / speedup	
EAGLE-3	1.99±0.01	1.00×	1.14±0.02	1.00×	0.87±0.01	1.00×	1.47±0.02	1.00×
TraceDraft	**2.22±0.02**	**1.10×**	**1.28±0.01**	**1.08×**	**0.93±0.01**	**1.06×**	**1.66±0.01**	**1.13×**
OSD-EAGLE-3	2.45±0.02	1.14×	1.50±0.01	1.10×	0.92±0.01	1.12×	1.73±0.02	1.67×
OSD-TraceDraft	**2.61±0.03**	**1.22×**	**1.59±0.02**	**1.17×**	**0.98±0.005**	**1.20×**	**1.84±0.02**	**1.78×**
Ens-EAGLE-3	2.52±0.02	1.16×	1.54±0.01	1.18×	0.94±0.01	1.14×	1.80±0.01	1.69×
Ens-TraceDraft	**2.67±0.03**	**1.23×**	**1.69±0.04**	**1.30×**	**0.99±0.005**	**1.22×**	**1.83±0.02**	**1.73×**
Datasets	**MT-Bench**		**PubMed**		**PubMedQA**		**TheoremQA**	
Method	avg len / speedup		avg len / speedup		avg len / speedup		avg len / speedup	
Target model: Vicuna-13B-v1.3								
EAGLE-3	4.98±0.02	1.00×	4.45±0.01	1.00×	5.00±0.01	1.00×	4.59±0.01	1.00×
TraceDraft	**5.23±0.02**	**1.13×**	**4.66±0.03**	**1.06×**	**5.26±0.02**	**1.05×**	**4.78±0.01**	**1.09×**
OSD-EAGLE-3	5.23±0.01	1.06×	4.77±0.02	1.09×	5.26±0.02	1.07×	4.80±0.02	1.11×
OSD-TraceDraft	**5.24±0.01**	**1.11×**	**4.80±0.02**	**1.11×**	**5.29±0.03**	**1.08×**	**4.82±0.02**	**1.11×**
Ens-EAGLE-3	5.35±0.03	1.15×	4.83±0.03	1.11×	5.37±0.02	1.10×	4.89±0.01	1.12×
Ens-TraceDraft	**5.36±0.04**	**1.17×**	**4.87±0.02**	**1.13×**	**5.38±0.03**	**1.12×**	**4.92±0.03**	**1.15×**
Datasets	**AIME2024**		**AIME2025**		**LiveCodeBench**		**LongBench-Write**	
Method	avg len / speedup		avg len / speedup		avg len / speedup		avg len / speedup	
EAGLE-3	4.64±0.02	1.00×	4.49±0.02	1.00×	4.55±0.02	1.00×	3.78±0.01	1.00×
TraceDraft	**4.88±0.04**	**1.10×**	**4.72±0.01**	**1.18×**	**4.82±0.02**	**1.02×**	**3.97±0.01**	**1.06×**
OSD-EAGLE-3	4.86±0.01	1.08×	4.76±0.01	1.21×	4.78±0.01	1.08×	3.97±0.01	1.07×
OSD-TraceDraft	**4.96±0.02**	**1.13×**	**4.81±0.03**	**1.23×**	**4.80±0.02**	**1.08×**	**4.02±0.02**	**1.10×**
Ens-EAGLE-3	5.00±0.01	1.16×	4.93±0.02	1.25×	4.89±0.03	1.10×	4.13±0.01	1.12×
Ens-TraceDraft	**5.05±0.02**	**1.19×**	**4.96±0.01**	**1.28×**	**4.90±0.02**	**1.15×**	**4.15±0.02**	**1.16×**

3 EXPERIMENTS

3.1 EXPERIMENT SETUP

3.1.1 DATASETS, EVALUATION AND MODELS

We evaluate eight workloads grouped by output budget: MT-Bench, PubMed, PubMedQA, and TheoremQA (short); AIME2024/2025, hard LiveCodeBench v6, and LongBench-Write (long). Targets are DeepSeek-R1-Distill-Llama-8B and Vicuna-13B-v1.3 with matching EAGLE-3 drafts; ablations use DeepSeek. Dataset references, subsets, and model checkpoints appear in Appendix A.

For K verification rounds, let a_k be accepted draft children, b_k the evaluated candidate-path width, and N the returned token count. We report

$$L = \frac{\sum_k a_k}{K}, \qquad a = 100\frac{\sum_k a_k}{\sum_k b_k}, \qquad V = \frac{N}{\mathcal{T}}, \qquad s = \frac{V_{\text{method}}}{V_{\text{EAGLE-3}}}. \tag{11}$$

Accepted length L excludes the root; including it gives approximately $L + 1$, subject to EOS/truncation. Speedups use matched baselines. $\mathcal{T}$ counts inference and all method-specific costs once, including supervision, training, checks, publication, Reset restoration, and outer training/merging (Appendix A). Cross-workload gains are macro-averaged.

3.1.2 Baselines

Baselines are fixed **EAGLE-3**, **OSD-EAGLE-3** (OnlineSPEC's single-learner implementation), and **Ens-EAGLE-3** (Liu et al., 2024; Qian et al., 2026).

3.2 Implementation Details

We use greedy decoding at batch size one on A800 GPUs. Main fixed-draft/Reset/Persistent configurations have five runs; all others have three. Results are means, with standard deviations where shown.

TraceDraft attempts one update after five verification rounds per selected turn; outer training occurs every 20 short-workload or five long-workload conversations. AIME2025 uses an 8,192-token generation cap in both standalone and composed runs. Appendix A gives full settings and controls.

Main/composed TraceDraft runs use Reset, initialized from the current outer checkpoint in compositions. Persistent tests state reuse. Each composition shares its baseline's evaluator and training checks.

We discuss TTS as a closely related arXiv preprint (Kumar et al., 2026). Our search of its paper and author pages on September 26, 2026 did not identify an official TTS implementation or a complete reproduction package; we therefore omit a matched experimental comparison.

4 Results

4.1 Main Results

Table 1 reports accepted length and end-to-end speedup relative to fixed EAGLE-3 within each target model. Figure 2 summarizes the main-model and domain-stream results.

(a) Within-response gains. On DeepSeek, Reset improves all eight workloads: mean accepted-length/throughput gains are **4.33%/2.34%** on short outputs and **10.84%/9.16%** on long outputs. Longer continuations offer more scope to amortize updates (Theorem 1), although content and budgets also differ.

(b) Composition and isolation. Against their own outer baselines, long-output OSD and ensemble combinations add **6.57–7.24%** and **2.54–10.03%** throughput; short-output gains are about 1–4%. End-to-end speedup also reflects the cost per verification round and method-specific overhead, so it need not scale proportionally with accepted length. PubMed, PubMedQA, and TheoremQA checks yield identical output tokens and outer checkpoints within each pair, supporting Proposition 2.

(c) Model transfer. Vicuna also gains accepted length and throughput on every workload. Its combinations increase accepted length and match or exceed their respective baselines' throughput.

4.2 Freezing and Update Frequency

We compare P-once and repeated Persistent updates (full results in Appendix B, Table 3).

(a) Reusable calibration. On PubMedQA, P-once freezes after the first successful update on the first request. Across the remaining 499 turns, it raises length from 3.7155 to 3.90 and throughput

from 91.4816 to 94.92 tok/s; continued P gives 3.91 and 94.38 tok/s. Early calibration remains useful without further training.

(b) Workload-dependent scheduling. Short-workload mean speedup rises from 1.014× ($f = 1$) to 1.039× ($f = 8, 32$). We use $f = 1$ as a common scheduling reference to evaluate conversation-local adaptation with Reset and state reuse with Persistent. P-once and sparse updates then assess the value of subsequent optimization. Their strong results show that early calibration can remain useful across requests, allowing recurring training to be reduced. Further updates still help on some workloads: on AIME2025, Persistent at $f = 1$ increases accepted length from P-once's 1.21 to 1.29 and speedup from 1.06× to 1.09×. These findings support choosing the update schedule according to the workload: continued adaptation is worthwhile when its decoding savings exceed its additional cost, as Equation (8) expresses.

(c) Early calibration and state retention. Reset and Persistent reach similar accepted lengths on the four shorter-output workloads. Both update only the output normalization and prediction head. The freezing and sparse-update results suggest that early calibration supplies much of this gain: Reset learns it afresh for each conversation, whereas Persistent retains it. Similar within-domain requests may require similar adjustments, leaving little benefit from further optimization. Domain switching tests whether that retained state also helps when incoming requests change.

4.3 Domain Switching

Table 2 reports five stream orders with matched baselines; AIME/Write denote AIME2024/LongBench-Write. The final panel reports individual domains.

Table 2: Domain switching: full streams and individual domains.

Domain Orders	AIME → Write				Write → AIME			
	Full stream		After switch		Full stream		After switch	
Method	avg len	speedup	avg len	speedup	avg len	speedup	avg len	speedup
EAGLE-3	1.28±0.03	1.00×	1.56±0.01	1.00×	1.28±0.02	1.00×	1.27±0.02	1.00×
TraceDraft Reset	1.53±0.01	1.12×	1.61±0.02	1.01×	1.35±0.02	1.06×	1.34±0.02	1.06×
TraceDraft P	**1.61±0.02**	**1.21×**	**1.64±0.03**	**1.04×**	**1.40±0.01**	**1.12×**	**1.39±0.01**	**1.12×**
Domain Orders	AIME → TheoremQA				TheoremQA → AIME			
	Full stream		After switch		Full stream		After switch	
Method	avg len	speedup	avg len	speedup	avg len	speedup	avg len	speedup
EAGLE-3	1.21±0.02	1.00×	1.20±0.02	1.00×	1.21±0.01	1.00×	1.22±0.01	1.00×
TraceDraft Reset	1.26±0.01	1.05×	1.23±0.01	1.04×	1.38±0.02	1.12×	1.34±0.01	1.09×
TraceDraft P	**1.32±0.02**	**1.08×**	**1.25±0.02**	**1.05×**	**1.24±0.02**	**1.02×**	**1.27±0.03**	**1.04×**
	AIME → TheoremQA → Write → PubMedQA							
Domain	AIME		TheoremQA		Write		PubMedQA	
Method	avg len	speedup	avg len	speedup	avg len	speedup	avg len	speedup
EAGLE-3	1.22±0.02	1.00×	1.20±0.01	1.00×	1.64±0.02	1.00×	3.77±0.02	1.00×
TraceDraft Reset	1.29±0.02	1.10×	1.23±0.01	1.08×	1.64±0.02	1.04×	3.94±0.01	1.10×
TraceDraft P	**1.38±0.01**	**1.15×**	**1.25±0.02**	**1.08×**	**1.67±0.01**	**1.06×**	**3.96±0.02**	**1.10×**

(a) Cross-domain reuse. Persistent reaches **1.21×/1.12×** full-stream speedup on AIME→Write and its reverse, and improves every four-domain segment over the fixed draft.

(b) Request compatibility. TheoremQA→AIME favors Reset: 1.12× versus P's 1.02× overall, and 1.09× versus 1.04× after switching. The final PubMedQA segment slightly favors Reset's throughput but P's length. Reuse depends on compatibility (Equation (6)); subsets, order, and mode-specific learning rates may contribute.

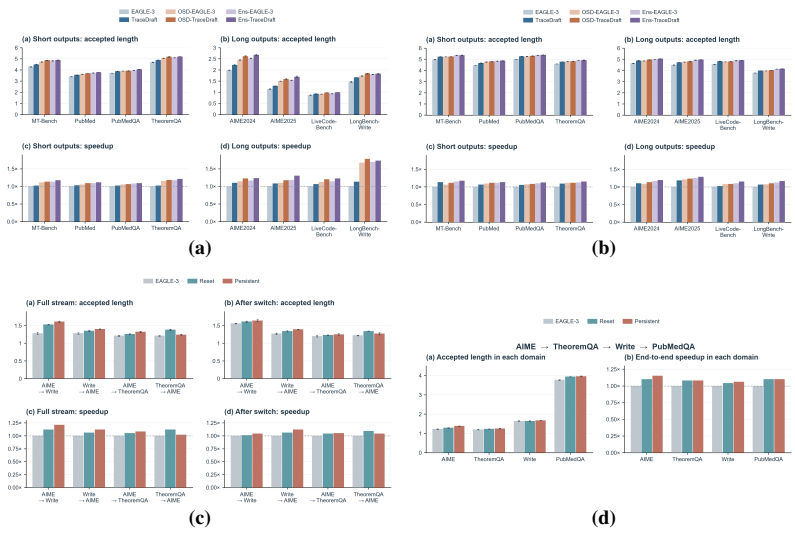

Figure 2: Results across models and request streams. (a) Main results with DeepSeek-R1-Distill-Llama-8B. (b) Main results with Vicuna-13B-v1.3. (c) Two-domain switching results with the DeepSeek target. (d) Four-domain streaming results with the DeepSeek target.

4.4 Ablation Study

Offline reference. Root-aligned online/offline gain ratios are about 70% on MT-Bench and 9–12% on PubMed/PubMedQA. Different evaluators and budgets make SFT a scale reference, not a ceiling (Appendix C).

Prefix source. AIME2024's 20-request audit confirms causal, aligned supervision, with 14/12 current/prior updates and 1,696/1,287 positions. Unequal budgets make the comparison descriptive (Appendix D).

5 Related Work

Drafting advances include EAGLE's feature prediction (Li et al., 2024a;b), vocabulary reduction (Zhao et al., 2025), and offline specialization (Hong et al., 2025). EAGLE-3 instead predicts tokens directly from fused target features (Li et al., 2025). Online adaptation spans OSD's query distillation, Aurora's asynchronous training, EvoSpec's vocabulary/parameter updates, and OnlineSPEC's optimistic/ensemble learning (Liu et al., 2024; Wang et al., 2026; Zhang et al., 2026; Qian et al., 2026). TTS studies within-response distillation and strided updates (Kumar et al., 2026). TraceDraft combines committed-prefix supervision, a fixed backbone, controlled reuse, and isolated outer-learner composition.

6 Discussion and Conclusion

TraceDraft adapts EAGLE-3's output layers from verified prefixes while preserving its backbone and committed-prefix caches. The long-output results support amortizing a useful early update over subsequent generation. Freezing keeps an adapted head useful without further training, while sparse updates retain gains with less frequent optimization. The OSD and OnlineSPEC compositions show additional gains over blockwise adaptation. Across changing domains, Persistent can reuse calibration through successive transitions, although Reset achieves higher throughput on some tested orders.

The evaluation covers two target models, each held fixed during adaptation, under synchronous single-request decoding. Cost-aware scheduling for batched serving is a natural extension. By preserving reusable computation and controlling each update's lifetime, TraceDraft provides a practical framework for request-time adaptation in EAGLE-3 decoding.

AI USE STATEMENT

Generative AI tools were used only for language polishing, prose revision, and the formatting of tables and references. The methodology, theoretical development, proofs, and experimental design were developed and conducted by the authors without substantive AI assistance. All AI-assisted content was reviewed and verified by the authors, who take full responsibility for the paper's text, technical claims, proofs, and experimental results.

ETHICS STATEMENT

The experiments use public benchmark requests. Persistent adaptation allows one conversation to influence the draft used for later conversations. A deployed service would need an explicit policy for user isolation and retention of learned state. Target verification governs the committed output, but does not itself provide isolation between users' adaptation states.

REPRODUCIBILITY STATEMENT

Algorithm 1 specifies the adaptation sequence. Section 3.2 and Appendix A give update, decoding, and outer-training settings; Section 4 distinguishes the comparison protocols. The theoretical results in Section 2 include their assumptions and proofs. Appendices B–D provide the complete schedule, offline-reference, and prefix-source results.

REFERENCES

Yushi Bai, Jiajie Zhang, Xin Lv, Linzhi Zheng, Siqi Zhu, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. LongWriter: Unleashing 10,000+ Word Generation from Long Context LLMs. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/59f278de1619bdb6b53fd04e8e0976e0-Abstract-Conference.html.

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*, 2023a. URL https://arxiv.org/abs/2302.01318.

Wenhu Chen, Ming Yin, Max Ku, Pan Lu, Yixin Wan, Xueguang Ma, Jianyu Xu, Xinyi Wang, and Tony Xia. TheoremQA: A theorem-driven question answering dataset. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7889–7901. Association for Computational Linguistics, 2023b. doi: 10.18653/v1/2023.emnlp-main.489. URL https://aclanthology.org/2023.emnlp-main.489/.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pp. 615–621. Association for Computational Linguistics, 2018. doi: 10.18653/v1/N18-2097. URL https://aclanthology.org/N18-2097/.

DeepSeek-AI. DeepSeek-R1-Distill-Llama-8B: Model card. Hugging Face model repository, 2025. URL https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Llama-8B. Accessed September 18, 2026.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the Knowledge in a Neural Network. *arXiv preprint arXiv:1503.02531*, 2015. URL https://arxiv.org/abs/1503.02531.

Fenglu Hong, Ravi Raju, Jonathan Lingjie Li, Bo Li, Urmish Thakker, Avinash Ravichandran, Swayambhoo Jain, and Changran Hu. Training Domain Draft Models for Speculative Decoding: Best Practices and Insights. *arXiv preprint arXiv:2503.07807*, 2025. URL https://arxiv.org/abs/2503.07807. Presented at the SCOPE Workshop at ICLR 2025.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. LiveCodeBench: Holistic and Contamination Free Evaluation of Large Language Models for Code. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/94074dd5a072d28ff75a76dabed43767-Abstract-Conference.html.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. PubMedQA: A dataset for biomedical research question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 2567–2577. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1259. URL https://aclanthology.org/D19-1259/.

Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *The Third International Conference on Learning Representations*, 2015. URL https://arxiv.org/abs/1412.6980.

Avinash Kumar, Sujay Sanghavi, and Poulami Das. Test-Time Speculation. *arXiv preprint arXiv:2605.09329*, 2026. URL https://arxiv.org/abs/2605.09329v2. Version 2.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast Inference from Transformers via Speculative Decoding. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 19274–19286. PMLR, 2023. URL https://proceedings.mlr.press/v202/leviathan23a.html.

Xuhong Li, Yves Grandvalet, and Franck Davoine. Explicit Inductive Bias for Transfer Learning with Convolutional Networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2825–2834. PMLR, 2018. URL https://proceedings.mlr.press/v80/li18a.html.

Yuhui Li. EAGLE3-DeepSeek-R1-Distill-LLaMA-8B: Model card. Hugging Face model repository, 2025a. URL https://huggingface.co/yuhuili/EAGLE3-DeepSeek-R1-Distill-LLaMA-8B. Accessed September 18, 2026.

Yuhui Li. EAGLE3-Vicuna1.3-13B: Model card. Hugging Face model repository, 2025b. URL https://huggingface.co/yuhuili/EAGLE3-Vicuna1.3-13B. Accessed September 26, 2026.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. EAGLE: Speculative Sampling Requires Rethinking Feature Uncertainty. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 28935–28948. PMLR, 2024a. URL https://proceedings.mlr.press/v235/li24bt.html.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. EAGLE-2: Faster inference of language models with dynamic draft trees. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 7421–7432. Association for Computational Linguistics, 2024b. doi: 10.18653/v1/2024.emnlp-main.422. URL https://aclanthology.org/2024.emnlp-main.422/.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. EAGLE-3: Scaling up Inference Acceleration of Large Language Models via Training-Time Test. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/c7b5a35ea98b62512a869c19ea7b03cb-Abstract-Conference.html.

Xiaoxuan Liu, Lanxiang Hu, Peter Bailis, Alvin Cheung, Zhijie Deng, Ion Stoica, and Hao Zhang. Online Speculative Decoding. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 31131–31146. PMLR, 2024. URL https://proceedings.mlr.press/v235/liu24y.html.

LMSYS. Vicuna-13B-v1.3: Model card. Hugging Face model repository, 2023. URL https://huggingface.co/lmsys/vicuna-13b-v1.3. Accessed September 26, 2026.

Mathematical Association of America. American Invitational Mathematics Examination (AIME). MAA invitational mathematics competition, 2025. URL https://maa.org/maa-invitational-competitions/. 2024 and 2025 examinations.

Yu-Yang Qian, Hao-Cong Wu, Yichao Fu, Hao Zhang, and Peng Zhao. When Drafts Evolve: Speculative Decoding Meets Online Learning. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://icml.cc/virtual/2026/poster/63017.

Junxiong Wang, Fengxiang Bie, Jisen Li, Zhongzhu Zhou, Zelei Shao, Yubo Wang, Yinghui Liu, Qingyang Wu, Avner May, Sri Yanamandra, Ce Zhang, Tri Dao, Percy Liang, Ben Athiwaratkun, Shuaiwen Leon Song, Chenfeng Xu, and Xiaoxia Wu. When RL Meets Adaptive Speculative Training: A Unified Training-Serving System. *arXiv preprint arXiv:2602.06932*, 2026. URL https://arxiv.org/abs/2602.06932v5. Version 5; accepted at ICML 2026.

Shuyu Zhang, Lingfeng Pan, Qicheng Wang, Yaqi Shi, Yueyang Tan, Ruyu Yan, Jiaqi Chen, Lixing Du, and Lu Wang. EvoSpec: Evolving Speculative Decoding via Real-Time Vocabulary and Parameter Adaptation. *arXiv preprint arXiv:2605.27390*, 2026. URL https://arxiv.org/abs/2605.27390.

Weilin Zhao, Tengyu Pan, Xu Han, Yudi Zhang, Ao Sun, Yuxiang Huang, Kaihuo Zhang, Weilun Zhao, Yuxuan Li, Jie Zhou, Hao Zhou, Jianyong Wang, Zhiyuan Liu, and Maosong Sun. FR-spec: Accelerating large-vocabulary language models via frequency-ranked speculative sampling. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3909–3921. Association for Computational Linguistics, 2025. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.198. URL https://aclanthology.org/2025.acl-long.198/.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/91f18a1287b398d378ef22505bf41832-Abstract-Datasets_and_Benchmarks.html.

A Detailed Experimental Configurations

The configurations below supplement Section 3.2.

We evaluate eight workloads. The shorter-output group contains MT-Bench conversations (Zheng et al., 2023), PubMed summarization (Cohan et al., 2018), PubMedQA (Jin et al., 2019), and TheoremQA (Chen et al., 2023b), with 80, 80, 500, and 800 requests, respectively. The long-output group contains the 30-question AIME2024 and AIME2025 examinations (Mathematical Association of America, 2025), 80 requests from a hard subset of LiveCodeBench v6 (Jain et al., 2025), and 80 LongBench-Write (2025) writing requests (Bai et al., 2025). Grouping refers to output budgets, not input length. The request counts above describe the DeepSeek evaluation. For Vicuna, we use 80 requests from each of the four shorter-output workloads. Unless noted below, Vicuna follows the corresponding DeepSeek settings.

We evaluate DeepSeek-R1-Distill-Llama-8B with its pretrained EAGLE-3 draft (DeepSeek-AI, 2025; Li, 2025a), and Vicuna-13B-v1.3 with EAGLE3-Vicuna1.3-13B (LMSYS, 2023; Li, 2025b). Table 1 reports both model pairs. The freezing, frequency, domain-switching, and other ablation experiments use DeepSeek.

The denominator $\mathcal{T}$ covers inference and all method-specific overhead for the evaluated workload. It includes prefill, decoding, supervision construction, feature recomputation, online optimization and checks, weight publication, Reset restoration, and outer training, checkpoint handling, and ensemble merging where applicable. Work outside the generation call is included, with each elapsed interval counted once. All reported throughput and speedup values use this complete timing scope. Cross-workload summaries are arithmetic means of per-workload relative gains, with equal weight per workload.

EAGLE-3 uses the pretrained draft with online updates disabled. **OSD-EAGLE-3** adapts OSD to EAGLE-3 using the single-learner training pipeline released with OnlineSPEC (Liu et al., 2024; Qian et al., 2026). **Ens-EAGLE-3** uses OnlineSPEC's ensemble pipeline (Qian et al., 2026). Each baseline and its TraceDraft combination share an implementation. Our adaptations retain actual generated token IDs, use a common evaluator, and add held-out update checks; these choices are shared across each pair.

We run all experiments with greedy, synchronous decoding at batch size one on NVIDIA A800 GPUs. The draft tree uses 60 tokens, depth 5, and top-$k = 10$. We repeat each configuration in the main comparison of fixed EAGLE-3, Reset, and Persistent five times. All other experiments are repeated three times per configuration. We report means over the corresponding runs; where shown, $\pm$ denotes the standard deviation across runs.

- **Generation budgets.** For DeepSeek, short-output workloads use a 1,024-token cap; AIME2024 and AIME2025 use 8,192 in both standalone and composed runs, and LiveCodeBench and LongBench-Write use 16,384. For Vicuna, MT-Bench, PubMed, and PubMedQA use 1,024 tokens, and TheoremQA uses 1,536.

- **TraceDraft updates.** In the DeepSeek experiments, we use one optimizer step after five verification rounds in each selected turn, $T = 1$, $\alpha = 0.25$, gradient clipping at 0.2, and maximum relative parameter step 0.02. Standalone long-output runs sample up to 128 positions from each source; composed runs use 64. Standalone short-output main comparisons and Persistent ablations use 10^{-6}. Reset uses 10^{-6} on AIME2024 and LiveCodeBench, and 10^{-5} on AIME2025, LongBench-Write, and composed runs. Held-out KL is a diagnostic in the original evaluator and an additional commit check in the composed implementation.

- **Outer training.** In the DeepSeek experiments, we train every 20 completed conversations on the shorter-output workloads and every five on the long-output workloads, skipping a final update with no following requests. Short-workload compositions use the same request counts as their corresponding baseline evaluations. OSD uses learning rate 10^{-5}. The ensemble uses three learners with rates $\{10^{-4}, 2 \times 10^{-4}, 4 \times 10^{-4}\}$, the `hedge` configuration, temperature 0.2, and FP32 merging. Both use two epochs, batch size two, seven-step recursive prediction, and loss decay 0.8.

- **Reuse experiments.** P-once freezes after its first successful commit. Sparse Persistent uses $f \in \{1, 8, 32\}$ on short tasks and $f \in \{1, 8, 16\}$ on all four long-output workloads.
- **Domain switching.** We compare fixed EAGLE-3, Reset, and Persistent on each ordered request stream. The AIME–Write experiments concatenate 20 AIME requests and 100 writing requests in either order, using pools separate from the standalone evaluation. We add AIME→TheoremQA, the reverse order, and AIME→TheoremQA→Write→PubMedQA. These three streams use the first 20 questions from each source, giving 40, 40, and 80 requests. The supplementary runs use an A800, a 16,384-token generation cap, a 32,768-token context limit, and learning rates 10^{-5} for Reset and 10^{-6} for Persistent. Persistent retains learner state through every domain boundary; Reset restores it at each conversation.
- **Controls.** In the AIME2024 prefix-source control, we use the current committed prefix at round five or shuffle among earlier completed requests with an independent seed of zero. Both branches skip the first request. Each successful update takes one optimizer step; draft features and teacher outputs are recomputed from the same prefix at aligned positions. Offline gains use the epoch-199 SFT checkpoint and its evaluator's baseline.

B Freezing and Update Frequency: Complete Results

Table 3 retains all eight workloads and all tested intervals. The TraceDraft row repeats the Reset results from Table 1; speedups use fixed EAGLE-3.

Table 3: Freezing and sparse updates.

Datasets	**MT-Bench**		**PubMed**		**PubMedQA**		**TheoremQA**	
Method	avg len	speedup	avg len	speedup	avg len	speedup	avg len	speedup
TraceDraft	**4.49±0.04**	**1.02×**	**3.55±0.02**	**1.03×**	**3.88±0.02**	**1.02×**	**4.88±0.03**	**1.02×**
P-once	4.51±0.02	1.05×	3.58±0.02	1.04×	3.90±0.01	1.04×	4.91±0.02	1.04×
P ($f = 1$)	4.49±0.02	1.02×	3.54±0.01	1.00×	3.88±0.02	1.02×	4.88±0.02	1.00×
P ($f = 8$)	4.51±0.01	1.06×	3.55±0.03	1.02×	3.92±0.02	1.04×	4.90±0.02	1.04×
P ($f = 32$)	4.50±0.03	1.03×	3.57±0.03	1.05×	3.94±0.02	1.05×	4.91±0.03	1.03×
Datasets	**AIME2024**		**AIME2025**		**LiveCodeBench**		**LongBench-Write**	
Method	avg len	speedup	avg len	speedup	avg len	speedup	avg len	speedup
TraceDraft	**2.22±0.02**	**1.10×**	**1.28±0.01**	**1.08×**	**0.93±0.01**	**1.06×**	**1.66±0.01**	**1.13×**
P-once	2.24±0.01	1.11×	1.21±0.02	1.06×	0.94±0.005	1.08×	1.69±0.03	1.13×
P ($f = 1$)	2.23±0.01	1.11×	1.29±0.02	1.09×	0.92±0.01	1.06×	1.68±0.01	1.12×
P ($f = 8$)	2.19±0.03	1.08×	1.28±0.02	1.08×	0.95±0.01	1.04×	1.69±0.02	1.14×
P ($f = 16$)	2.28±0.02	1.12×	1.32±0.01	1.11×	0.94±0.01	1.04×	1.70±0.02	1.15×

C Offline Reference

Offline reference. We compare the scale of online and offline improvement in Table 4. Let A_{SFT} and $A_{\mathrm{off},0}$ be archived root-inclusive offline lengths. We use

$$g_{\mathrm{off}} = \frac{A_{\mathrm{SFT}}}{A_{\mathrm{off},0}} - 1, \qquad g_{\mathrm{on}} = \frac{L_{\mathrm{on}} + 1}{L_0 + 1} - 1, \qquad \rho = \frac{g_{\mathrm{on}}}{g_{\mathrm{off}}}. \tag{12}$$

The evaluators and training budgets differ, so ρ describes relative gain scales under their respective baselines. On MT-Bench, online-to-offline gain ratios are approximately 70%. They are 9–12% on PubMed and PubMedQA, where offline gains are larger. A small amount of request-time training thus recovers useful agreement, while more extensive offline specialization can offer additional gains. The SFT checkpoint is a scale reference, not an optimization ceiling.

Table 4: Online and offline relative gains on overlapping workloads.

Method / statistic	MT-Bench	PubMed	PubMedQA
Offline SFT: $100g_{\text{off}}$	5.58	33.36	29.58
Reset: $100g_{\text{on}}$	3.96	3.14	3.52
Reset: 100ρ	71.00	9.40	11.90
Persistent: $100g_{\text{on}}$	3.90	2.87	3.46
Persistent: 100ρ	69.90	8.60	11.70

D PREFIX-SOURCE AGGREGATES

Table 5 reports the current-request and earlier-request prefix results on AIME2024. Across 20 requests, the two branches perform 14 and 12 updates using 1,696 and 1,287 valid supervision positions, respectively. These unequal update budgets support a descriptive comparison, not a matched-budget attribution of gains to the prefix source.

Table 5: Current-request and earlier-request prefix supervision.

	AIME2024	
Prefix source	avg len	tok/s
Current request	2.13±0.03	52.40
Earlier request	2.15±0.05	51.89