CAN COMPUTATION FROM EARLIER PROBLEMS HELP LLMS SOLVE NEW ONES?

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models often solve independent problems in the same conversation. Can computation from earlier problems help them solve new ones? To answer this question, we first conduct preliminary experiments showing that retained history can raise or lower later-turn accuracy, even within the same domain. To understand these effects, we use controlled replay to isolate internal state changes specific to each problem–history pairing. Across different histories, these changes preserve similar relationships among current problems. To improve reasoning under retained history, we introduce **STAIR** (**Stale-Token Attention for Inter-query Reuse**). STAIR captures keys and values from earlier response generation in a fixed bank. It learns to redirect current queries when they read this bank during prompt processing. The base model remains frozen; only 12,288 parameters are trained. Across three Qwen models and four benchmarks, STAIR improves average later-turn accuracy by up to 11.67 percentage points over the unmodified model with history. Code is provided in the supplementary material.

1 INTRODUCTION

When a user asks a large language model (LLM) to solve one problem, and then poses another in the same session, the conversation carries forward the work on earlier questions. Each new problem is independent of the earlier ones and supplies its own task-specific information. For the user, the earlier turn remains visible as text in the conversation context. For the model, producing that text also involved a sequence of internal computations. Processing the retained conversation makes earlier token positions available to current queries through attention. Language models already reuse representations from earlier inputs (Dai et al., 2019; Wu et al., 2022). Whether earlier computation remains useful after a task switch is less clear. In this paper, we ask: ***Can the historical computation left by an earlier problem still help solve the current one?***

To examine how earlier problems affect later reasoning, we place the same problem set at different positions in four-turn sessions. The first turn (Vanilla) has no earlier problem in context; at later turns, the unmodified model (Native) retains earlier problems and assistant responses. We keep the sampling configuration and checkpoint fixed across turns. We report Avg@4, the average correctness over four sampled responses per problem. On MATH-500 (Lightman et al., 2024) with Qwen3-4B Instruct (Qwen Team, 2025), Avg@4 falls from 95.00% under Vanilla to 93.75% at Native T4. By contrast, on GPQA-Diamond (Rein et al., 2024) with Qwen3.5-4B (Qwen Team, 2026a), Avg@4 rises from 63.64% under Vanilla to 75.25% at each of Native T2, T3, and T4. Retained history can hurt or help, depending on the model and task.

To examine the computation behind these changes, we replay the retained conversations with Qwen3-4B Instruct's weights fixed. Attention to earlier assistant responses persists into middle and later layers, and the current problem's representation shifts with history. We separate the average effects associated with current problems and histories. The remaining changes retain similar nearest-neighbor relations across distinct histories, far above a shuffled-identity control. Paired Native and Vanilla answers to the same problem show both corrected errors and lost correct answers.

To learn which historical states to emphasize or suppress, we introduce **STAIR**, short for **S**tale-**T**oken **A**ttention for **I**nter-query **R**euse (Figure 1). Stale tokens are earlier assistant tokens whose original problem is complete. STAIR stores their captured attention keys and values (K/V) in a read-only bank. A small controller learns a direction for each query head. It reflects queries by reversing

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.

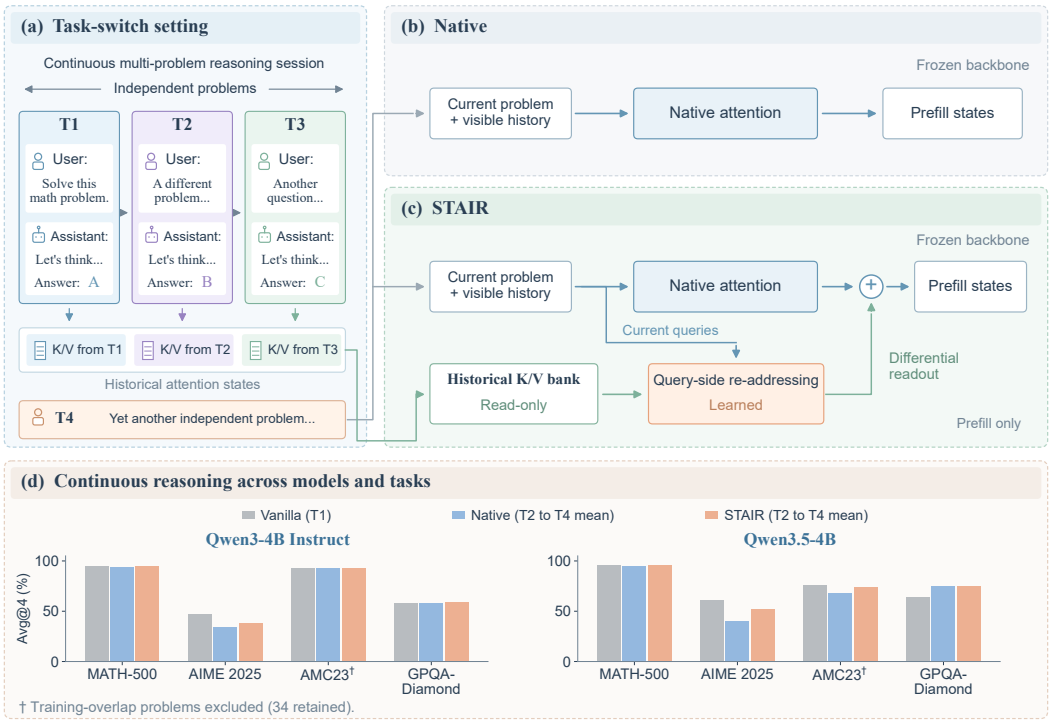

Figure 1: STAIR in a continuous problem-solving session. **(a)** Earlier problems leave conversation history; the assistant responses illustrate Instruct, where reasoning and answers remain visible. T4 introduces a new independent problem. **(b)** Native is the unmodified model reading that history. **(c)** STAIR changes how current queries read a separate bank of attention keys and values captured during earlier turns. It does so while processing the current prompt (prefill), with the model weights fixed. Both conditions process the visible conversation anew each turn. **(d)** Avg@4 averages correctness over four sampled answers per problem for the two 4B models. Vanilla uses T1; Native and STAIR entries average T2–T4. † AMC23 excludes training-overlap problems, leaving 34 problems.

their component along that direction, preserving their length. These reflections change the attention weights assigned to historical states, while model weights and stored entries stay fixed. STAIR acts during prompt processing (prefill); subsequent generation follows the original decoding path.

With 12,288 trainable parameters and a frozen backbone, STAIR improves mean T2–T4 Avg@4 on AIME 2025 (Zhang & Math-AI Team, 2025) by 11.67 percentage points for Qwen3.5-4B and 6.11 points for Qwen3.5-9B over Native.

To conclude, our contributions are:

1. We find a recurring, problem-dependent component in history-conditioned representation changes across distinct source histories.
2. We introduce STAIR, which learns query reflections to change how the current problem reads a fixed historical K/V bank while keeping the backbone frozen.
3. Across three Qwen models and four reasoning benchmarks, STAIR achieves gains of up to 11.67 percentage points in mean T2–T4 Avg@4 over Native while training only 12,288 parameters.

2 Historical Computation After Task Switches

To see how earlier turns affect a new problem, we hold that problem fixed and vary the preceding conversation. We call this preceding conversation the *source history*. Using Qwen3-4B Instruct on MATH-500, we replay each current problem after completed three-turn histories: the frozen model

processes the saved conversation text and current problem again. The current problem text stays the same across histories and never appears in its source history. We use two data groups with disjoint current problems and history sources (Figure 2A). Appendix A gives the replay and measurement details.

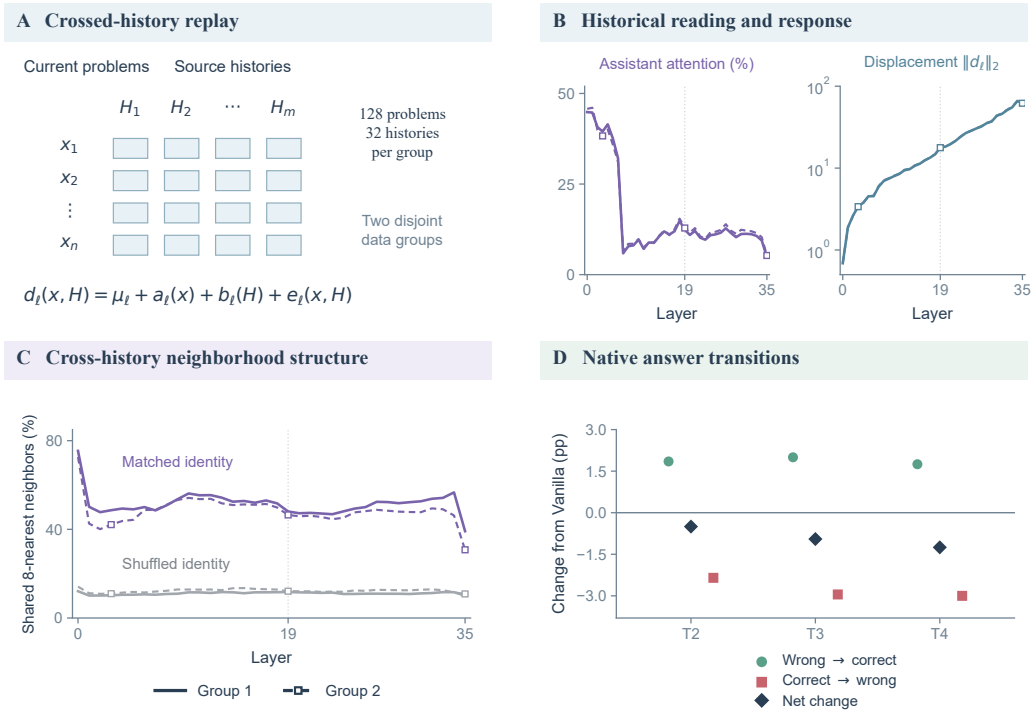

Figure 2: Historical computation after task switches in Qwen3-4B Instruct on MATH-500. **A:** Crossed-history replay with a frozen backbone. Each of two disjoint data groups crosses 128 current problems with 32 three-turn source histories. State displacement is decomposed into additive effects and an interaction response. **B:** Attention mass on earlier assistant-response bodies (left) and the norm of the current problem's state displacement (right; logarithmic scale). **C:** Cross-history overlap of eight nearest candidate problems in interaction-response space, with the Question-identity shuffle control. At layer 19, interaction energy fractions are 0.69% and 0.66% for groups 1 and 2. In B and C, solid lines denote group 1; dashed lines with hollow squares denote group 2. Vertical dotted lines mark layer 19, reported in the text. Layers are zero-indexed. **D:** Native answer transitions relative to Vanilla T1, normalized by 2,000 pairs per turn. Wrong-to-correct contributions are positive and correct-to-wrong contributions negative. Diamonds mark net changes.

2.1 Attention to Earlier Responses and State Displacement

To see how the model allocates attention after a task switch, we measure the share that queries from the current problem prompt assign to earlier assistant-response positions at each layer. We call this share *attention mass*. The earlier responses include reasoning and final answers. We also measure how history changes the current problem's representation. Let $h_\ell(x \mid H)$ be the output of block ℓ for problem x under history H, averaged over the current problem's body tokens. Its *state displacement* is

$$d_\ell(x, H) = h_\ell(x \mid H) - h_\ell(x \mid \varnothing). \tag{1}$$

The no-history condition retains the same system prompt and current-turn template.

Both data groups show similar layerwise patterns (Figure 2B). At layer 3, earlier assistant responses receive 39.49% and 38.29% of the attention mass, respectively. At layer 19, they still receive 12.41% and 12.87%. State displacement also persists into later layers.

2.2 Interaction Responses Retain Problem-Dependent Structure

Does a problem retain similar neighbors when its source history changes? Such consistency would suggest that the internal response to history has a recurring organization tied to the current problem.

A shift shared by all problems under one history could preserve their neighborhoods by moving them together. To isolate the change specific to a problem–history pairing, we decompose displacement over the balanced design:

$$d_\ell(x, H) = \mu_\ell + a_\ell(x) + b_\ell(H) + e_\ell(x, H), \tag{2}$$

Here μ_ℓ is the overall mean. The terms $a_\ell(x)$ and $b_\ell(H)$ capture changes associated with the problem and the history separately. The remainder $e_\ell(x, H)$ is the *interaction response*, which depends on their pairing. At layer 19, it accounts for 0.69% and 0.66% of total displacement energy in the two groups, measured by sums of squared norms.

For each current problem, we find its eight nearest candidate problems by Euclidean distance between interaction responses. We repeat this under two histories built from disjoint earlier problems. Cross-history neighborhood overlap measures how many neighbors retain the same identities. As a control, we shuffle which current problem anchors the second neighborhood, leaving the represented problems unchanged.

Neighborhood overlap reaches 48.06% and 46.50% in the two groups, compared with 11.73% and 12.08% under the shuffle control (Figure 2C). The interaction occupies a small fraction of total displacement energy, yet its local neighborhood structure recurs across histories.

2.3 Native Continuous Reasoning Changes Answers in Both Directions

To measure how retained history changes answer correctness, we pair each later Native response with its Vanilla T1 answer to the same problem and sample. Across 500 MATH-500 problems, this gives 2,000 pairs per turn. Later turns include the conversation generated at earlier turns.

At T4, 35 answers change from wrong to correct and 60 from correct to wrong (Figure 2D). Avg@4 falls by 1.25 percentage points. To examine the role of earlier answer correctness, we also group natural sessions by whether the first answer in the same session was correct. Appendix A.5 gives the breakdown.

Table 1: Accuracy (%) pooled over T2–T4 across three models and three mathematical benchmarks; n counts responses.

T1	n	Vanilla	Native	STAIR
Correct	18,879	93.96	93.42	**94.18**
Wrong	1,425	80.07	60.35	**68.56**

In this pooled mathematical comparison, Native falls below Vanilla even after correct T1 answers, and STAIR improves both groups. Current queries determine the attention weights assigned to earlier assistant states. Learning this query-to-history map offers a way to adjust the readout while keeping the stored states fixed.

3 STAIR Re-addresses Frozen Historical Computation

To control how the current problem uses earlier computation, STAIR learns how current queries address a separate, fixed historical K/V bank. Native and STAIR both process the visible conversation again at each turn. STAIR also reads states captured during earlier response generation (Figure 3). For Qwen3.5, the visible conversation contains earlier problems and final answers, while the separate bank holds K/V states from the reasoning that produced them. For Instruct, earlier assistant answers appear in the conversation and supply states to the bank. STAIR leaves the stored entries unchanged. Its additional read updates the current computation only while the model processes the new prompt, from the current user message through the assistant generation prefix.

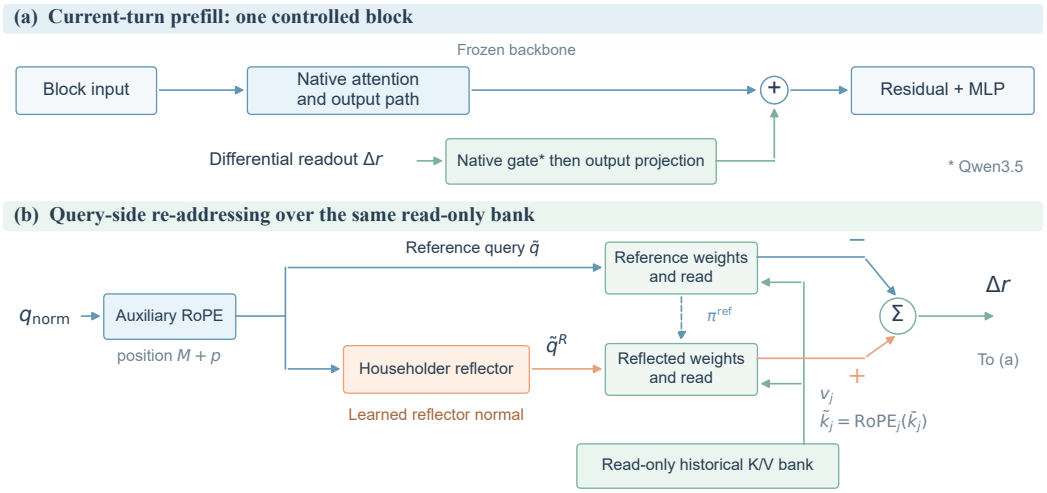

Figure 3: STAIR within a controlled block. (a) The change between two reads of the historical bank passes through the frozen output path and is added to the native attention output. Qwen3.5 also uses its native gate. (b) A learned reflection redirects the current query after auxiliary position encoding (RoPE). The original and reflected queries read the same fixed bank; subtracting their readouts gives Δr. Equation 7 specifies the reflected read. Only the reflection directions are learned.

3.1 A Read-only Historical K/V Bank

STAIR keeps earlier computation available in an ordered bank of attention keys and values. Keys determine where a query attends; values supply the content it reads. At turn t, for controlled layer ℓ and K/V head g, let

$$\mathcal{B}_{t,\ell,g} = \{(\bar{k}_{\ell,g,j}, v_{\ell,g,j})\}_{j=0}^{M_t-1}, \tag{3}$$

where M_t is the number of stored token positions. Keys are captured after key projection and normalization, before rotary position encoding (RoPE; Su et al., 2024); paired values are the corresponding value-projection outputs. RoPE rotates queries and keys according to their token positions.

The bank contains states from the reasoning body in Qwen3.5 models and the assistant-answer body in Instruct models. Bank entries are read-only throughout the current turn. New states are detached when captured and appended for later turns, preserving the entries already stored. Appendix B.1 specifies the capture boundaries.

The auxiliary branch assigns consecutive positions $0, \ldots, M_t - 1$ to bank keys. A current token at position p in the full conversation input, excluding left padding, receives auxiliary position $M_t + p$. The positioned query and keys are

$$\tilde{q} = \mathrm{RoPE}_{M_t+p}(q_{\mathrm{norm}}), \qquad \tilde{k}_j = \mathrm{RoPE}_j(\bar{k}_j), \tag{4}$$

where q_{norm} is the query after projection and normalization. The native attention path retains its original positions and masks.

3.2 Query-side Re-addressing

To change how a current query weights historical entries, STAIR reflects the query before its auxiliary bank read. The layerwise diagnostics in Section 2 motivated our choice of layers $[3, 11, 19]$ (zero-based), spanning early and intermediate computation. We kept this placement fixed across all three model configurations.

Each controlled query head learns a reflector normal $n_{\ell,h}$, a vector perpendicular to the reflection plane. Its Householder reflection (Householder, 1958) preserves query norm by reversing the component along this direction:

$$u_{\ell,h} = \frac{n_{\ell,h}}{\max(\|n_{\ell,h}\|_2, 10^{-8})}, \qquad \tilde{q}^R = \tilde{q} - 2u_{\ell,h}(u_{\ell,h}^\top \tilde{q}). \tag{5}$$

The orthogonal query component is unchanged. Reflection follows auxiliary RoPE. Its displacement depends on the current query, allowing a shared reflector to change the address differently for each token.

The original positioned query defines a smoothed reference distribution over the bank. Suppressing layer, head, and current-token indices, it is

$$\pi_j^{\mathrm{ref}} = (1-\varepsilon)\,\mathrm{softmax}_j\left(\frac{\tilde{q}^\top \tilde{k}_j}{\sqrt{d_h}}\right) + \frac{\varepsilon}{M_t}, \qquad \varepsilon = 10^{-6}, \tag{6}$$

where d_h is the head dimension. The reflected query changes the query–key scores. With $c = \tilde{q}^R - \tilde{q}$, its distribution reweights the smoothed reference:

$$\pi_j^R = \frac{\pi_j^{\mathrm{ref}} \exp(c^\top \tilde{k}_j/\sqrt{d_h})}{\sum_s \pi_s^{\mathrm{ref}} \exp(c^\top \tilde{k}_s/\sqrt{d_h})}. \tag{7}$$

Both distributions normalize over the historical K/V bank. They yield the reference read and reflected read,

$$r^{\mathrm{ref}} = \sum_j \pi_j^{\mathrm{ref}} v_j, \qquad r^R = \sum_j \pi_j^R v_j. \tag{8}$$

The controller determines where to read; the bank supplies the content at those addresses.

3.3 Differential Readout and Learning

To isolate what the reflection changes, STAIR subtracts the reference read from the reflected read:

$$\Delta r = r^R - r^{\mathrm{ref}} = \sum_j (\pi_j^R - \pi_j^{\mathrm{ref}}) v_j. \tag{9}$$

The difference weights sum to zero, so the differential readout captures the change in content induced by redistributing attention over the same bank.

After concatenation, head-wise readouts pass through the native sigmoid output gate in Qwen3.5 and then through the frozen attention output projection weight in both architectures. The projected update is added to the native self-attention output, and the block proceeds with residual addition and its MLP. After prefill, the auxiliary branch is disabled. Autoregressive decoding follows the original computation path with the modified prefix states, while newly produced historical attention states are captured for subsequent turns.

To learn useful reads while staying close to Native predictions, we train each controller on responses its frozen backbone generates for the DAPO-Math-17k dataset (Yu et al., 2025; BytedTsinghua-SIA, 2025). We retain correct and incorrect responses and arrange them into four-turn sessions. Native replay of T1 supplies the initial history. At T2–T4, teacher forcing feeds the stored response tokens back into the model. These turns supply supervision and new bank entries from the STAIR computation. The auxiliary branch acts on the current prompt suffix, as at inference. Response losses reach the controller through this prefix computation. Stored bank entries remain detached across turns.

For target token y and its text prefix s, the loss is

$$\ell(y, s) = -\log p_{\mathrm{STAIR}}(y \mid s) + \lambda D_{\mathrm{KL}}(p_{\mathrm{Native}}(\cdot \mid s) \,\|\, p_{\mathrm{STAIR}}(\cdot \mid s)), \qquad \lambda = 1. \tag{10}$$

The Native teacher independently processes the same history text, current prompt, and teacher-forced response prefix through the native attention path. Supervision covers the full stored responses at T2–T4. Token losses are averaged within each response and weighted to balance problems. Only the reflector normals receive parameter updates. Appendix B.4 details session replay and gradient flow; Appendix B.5 defines the loss aggregation.

4 Experiments

4.1 Experimental Setup

To test whether learned historical reading helps across model sizes and tasks, we study Qwen3-4B Instruct (Yang et al., 2025a; Qwen Team, 2025), Qwen3.5-4B, and Qwen3.5-9B (both in thinking

mode) (Qwen Team, 2026a;b). Each controller uses responses from its own backbone on DAPO-Math-17k, with 14,806 training and 128 validation problems. STAIR trains for one epoch, and we use the final scheduled checkpoint. Appendix C.1 gives the training settings.

Evaluation covers MATH-500 (Hendrycks et al., 2021; Lightman et al., 2024), AIME 2025 (Zhang & Math-AI Team, 2025), and AMC23† (Math-AI, 2025). GPQA-Diamond (Rein et al., 2024) tests out-of-domain transfer from the controller's mathematical training data to scientific reasoning. We compare STAIR with Native in sessions retaining earlier problems and assistant responses. Vanilla answers each problem without history. Native and STAIR share problem schedules and sample seeds, then continue from their own generated histories.

We sample four responses per problem at each turn. Avg@4 averages response correctness; Pass@4 measures the fraction of problems with at least one correct response. Appendix C gives session construction, decoding settings, and scoring.

4.2 Continuous Reasoning

We compare Native and STAIR over turns T2–T4 to measure performance after earlier problems accumulate (Table 2). Vanilla gives the no-history T1 reference. Appendix D reports each turn separately.

Table 2: Continuous reasoning results (%). Vanilla reports T1; Native and STAIR average T2–T4. Bold marks the better Native/STAIR score for each metric, including ties. Underlined STAIR scores exceed the corresponding Vanilla reference.

	MATH-500		AIME 2025		AMC23†		GPQA-Diamond	
Condition	Avg@4	Pass@4	Avg@4	Pass@4	Avg@4	Pass@4	Avg@4	Pass@4
Qwen3-4B Instruct								
Vanilla	95.00	97.20	47.50	66.67	92.65	97.06	58.59	78.79
Native	94.10	**97.60**	34.44	47.78	**93.38**	96.08	58.00	76.94
STAIR	**94.63**	97.53	**38.06**	**52.22**	**93.38**	**97.06**	**58.75**	**77.78**
Qwen3.5-4B								
Vanilla	96.00	98.60	60.83	80.00	76.47	85.29	63.64	75.76
Native	94.78	98.80	40.56	64.44	68.14	87.25	75.25	**89.39**
STAIR	**96.27**	**99.27**	**52.22**	**77.78**	**74.51**	**90.20**	**75.38**	86.87
Qwen3.5-9B								
Vanilla	96.65	98.20	61.67	73.33	77.94	85.29	68.81	80.30
Native	96.38	**99.20**	43.33	68.89	72.55	91.18	**79.88**	**89.73**
STAIR	**96.78**	99.13	**49.44**	**73.33**	**75.74**	**92.16**	75.93	88.55

† AMC23 excludes problems overlapping the training set, leaving 34 problems.

The largest gains over Native occur on AIME 2025. STAIR raises mean T2–T4 Avg@4 by 11.67 points for Qwen3.5-4B, 6.11 for Qwen3.5-9B, and 3.61 for Qwen3-4B Instruct. All three improve Avg@4 at each later turn and mean Pass@4 on this benchmark. On MATH-500, both Qwen3.5 models exceed Vanilla in both mean metrics. Qwen3.5-4B also gains 6.37 Avg@4 points on AMC23†. On GPQA-Diamond, Instruct improves both metrics, while the Qwen3.5 models show no comparable gain.

4.3 Historical Readout from a Shared T1 History

To isolate the current-turn read under a shared history, we evaluate Qwen3.5-4B on AIME 2025 at T2. All conditions use the same Native T1 responses, generated separately from Table 2. Replaying their saved tokens builds the historical K/V bank. We match the visible history, current problem, generation seed, and decoding budget.

To identify what matters in that read, we compare STAIR with three test-time interventions and a separately trained Bank-free controller (Table 3). Random-reflector replaces the learned reflection directions at test time; Direct reflected-read adds the full reflected read instead of its difference from the reference. The Bank-free controller adjusts the current attention output with 12,288 trainable parameters and no auxiliary bank read. It uses the same data and update budget. The K–V pairing control permutes historical values within each K/V head while keeping keys, positions, and the learned controller fixed.

Table 3: Shared-T1 AIME 2025 results for Qwen3.5-4B at T2 (%). Conditions share 30 problems and four matched trajectories per problem. Random-reflector and K–V pairing average three test-time seeds; other conditions use one evaluation. Δ is the Avg@4 difference from Native.

Condition	Avg@4	Δ (pp)	Pass@4
Native	21.67	0.00	56.67
Random-reflector	37.22	+15.56	63.33
Direct reflected-read	41.67	+20.00	73.33
Bank-free	36.67	+15.00	66.67
K–V pairing permutation	33.06	+11.39	61.11
STAIR	**48.33**	**+26.67**	**80.00**

With T1 held fixed, STAIR reaches 48.33% Avg@4 against 21.67% for Native, a 26.67-point gain; Pass@4 rises from 56.67% to 80.00%. The separately trained Bank-free controller reaches 36.67%, while disrupting the bank's K–V pairings lowers STAIR to 33.06% across three seeds. Random-reflector and Direct reflected-read also trail the full method (Table 3). Among these controls, STAIR's learned differential readout over intact K/V pairs yields the highest T2 accuracy. Appendix C.4 gives the shared-input protocol.

4.4 LoRA Comparison and Joint Training

To test whether learned historical access complements weight adaptation, we train STAIR jointly with low-rank adaptation (LoRA; Hu et al., 2022). Training data, objective, and update budget match across runs. For Qwen3-4B Instruct on AIME 2025, STAIR improves Avg@4 over Native by 3.61 points with 12,288 trainable parameters (Table 4). LoRA trains 336 times as many. Joint training adds 12,288 parameters and improves Avg@4 over LoRA by 3.89 points. Pass@4 rises by 4.44 points in both comparisons.

Table 4: Qwen3-4B Instruct on AIME 2025 (%). Scores average T2–T4. Δ is computed before rounding, relative to Native for STAIR and to LoRA for LoRA + STAIR. Bold marks the highest score and gain for each metric, including ties.

		Avg@4		Pass@4	
Condition	Trainable params	Score	Δ (pp)	Score	Δ (pp)
Native	0	34.44	–	47.78	–
STAIR	12,288	38.06	+3.61	52.22	**+4.44**
LoRA	4,128,768	39.44	–	58.89	–
LoRA + STAIR	4,141,056	**43.33**	**+3.89**	**63.33**	**+4.44**

4.5 Bank Prefix and Runtime

Limiting access to saved K/V tokens tests how performance changes with bank length. On the shared-T1 AIME 2025 inputs, the first 2K, 8K, or 32K tokens yield 35.00%, 43.33%, and 39.17% T2 Avg@4, compared with 48.33% when all saved tokens are available. In a separate fixed-length runtime test, full-bank reading adds a median 0.81 s and 6.60 GiB in peak memory over Native. Appendix C.4 gives the length protocol and cost breakdown.

5 Related Work

Transformer-XL and Recurrent Memory Transformer carry information across text segments through recurrence (Dai et al., 2019; Bulatov et al., 2022). Memorizing Transformers retrieve past key–value pairs by approximate nearest-neighbor search before attending over a non-differentiable memory (Wu et al., 2022). StreamingLLM retains initial attention-sink tokens alongside a recent-token window to sustain streaming inference (Xiao et al., 2024).

Repeated input text offers another opportunity for reuse. Prompt Cache precomputes attention states for reusable prompt modules (Gim et al., 2024). CacheBlend combines cached text chunks with selective recomputation to recover interactions with preceding text (Yao et al., 2025). Both reduce prefill work by reusing computation associated with text in the current input.

Prefix-tuning and soft prompt tuning learn continuous task-specific inputs while keeping the model frozen (Li & Liang, 2021; Lester et al., 2021). For reasoning, pause tokens allow additional computation before answer generation (Goyal et al., 2024). Coconut feeds the model's last hidden state back as the next input embedding, allowing successive reasoning steps in continuous latent space (Hao et al., 2025). A learned coprocessor can augment a frozen language model's cache with latent embeddings (Liu et al., 2025), while KV-derived representations can guide sampling and switching between fast and slow thinking (Xing et al., 2026). At the attention-operator level, Differential Transformer subtracts two attention maps with a learned coefficient before reading shared values (Ye et al., 2025). STAIR subtracts a reference read from a query-reflected read over the same historical bank during prefill, keeping the backbone and stored states fixed.

Laban et al. (2026) study conversations in which information about one task is supplied incrementally, finding that early assumptions can persist as further instructions arrive. Our sessions present a complete, independent problem at each turn.

6 Discussion and Limitations

The contrast between AIME 2025 and GPQA-Diamond raises a question about the transfer of learned reading rules. Reflector directions are fitted on mathematical problems, and their effect on attention depends on the current query and historical keys. One possibility is that these directions adapt to query–history relationships frequent in the training distribution. Broader training mixtures would test whether this dependence contributes to the observed domain variation. Current evidence covers four-turn sessions in the Qwen family.

Longer sessions increase bank storage and prefill reading costs as completed turns contribute new states. Prefill-only operation confines auxiliary reads to the start of each turn. Selective retention could limit bank growth as the conversation continues.

Pretraining on diverse task switches could learn reflector directions alongside the backbone. The joint LoRA result supports learning weight adaptation and historical access together. Supervised fine-tuning or reinforcement learning could further test how the readout adapts as reasoning behavior changes.

7 Conclusion

After a task switch, different histories leave a recurring problem-dependent pattern in how the current problem is represented. STAIR separately stores K/V states during earlier generation and learns how current queries read them while keeping the backbone fixed. Across four benchmarks on three Qwen models, it achieves gains of up to 11.67 percentage points in mean T2–T4 Avg@4 over Native while training only 12,288 parameters. These results support treating the computation left by completed problems as a resource whose readout can be learned for subsequent reasoning.

AI USE STATEMENT

Generative AI tools assisted with phrasing, grammar, and readability during manuscript preparation. The authors reviewed and revised AI-assisted text and take responsibility for the paper's claims and final manuscript.

ETHICS STATEMENT

The study evaluates offline four-turn sessions assembled from the cited public question sets. Earlier assistant responses are generated by the evaluated models; no human participants were recruited, and no private user conversations were collected.

REPRODUCIBILITY STATEMENT

Appendix A documents the replay design and representation measurements. Appendix B specifies state capture, query re-addressing, and training. Appendix C gives the data, model, prompt, decoding, and scoring protocols, including the shared-history controls. Appendix D reports per-turn results. The supplementary code archive contains scripts for rollout generation, session construction, training, evaluation, and scoring.

REFERENCES

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebron, and Sumit Sanghai. GQA: Training generalized multi-query transformer models from multi-head checkpoints. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 4895–4901, Singapore, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.298. URL https://aclanthology.org/2023.emnlp-main.298/.

Aydar Bulatov, Yury Kuratov, and Mikhail Burtsev. Recurrent memory transformer. In *Advances in Neural Information Processing Systems*, volume 35, pp. 11079–11091. Curran Associates, Inc., 2022. doi: 10.52202/068431-0805. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/47e288629a6996a17ce50b90a056a0e1-Abstract.html.

BytedTsinghua-SIA. DAPO-Math-17k. Hugging Face dataset, 2025. URL https://huggingface.co/datasets/BytedTsinghua-SIA/DAPO-Math-17k.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 2978–2988, 2019. doi: 10.18653/v1/P19-1285.

Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/98ed250b203d1ac6b24bbcf263e3d4a7-Abstract-Conference.html.

In Gim, Guojun Chen, Seung-seob Lee, Nikhil Sarda, Anurag Khandelwal, and Lin Zhong. Prompt Cache: Modular attention reuse for low-latency inference. In *Proceedings of Machine Learning and Systems*, volume 6, pp. 325–338, 2024. URL https://proceedings.mlsys.org/paper_files/paper/2024/hash/a66caa1703fe34705a4368c3014c1966-Abstract-Conference.html.

Sachin Goyal, Ziwei Ji, Ankit Singh Rawat, Aditya Krishna Menon, Sanjiv Kumar, and Vaishnavh Nagarajan. Think before you speak: Training language models with pause tokens. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=ph04CRkPdC.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason E. Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=Itxz7S4Ip3.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/be83ab3ecd0db773eb2dc1b0a17836a1-Abstract-round2.html.

Arthur E. Hoerl and Robert W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970. doi: 10.1080/00401706.1970.10488634.

Alston S. Householder. Unitary triangularization of a nonsymmetric matrix. *Journal of the ACM*, 5 (4):339–342, 1958. doi: 10.1145/320941.320947.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. LLMs get lost in multi-turn conversation. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/59f6421e64707225fdf5b28840679a07-Abstract-Conference.html.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3045–3059. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.emnlp-main.243. URL https://aclanthology.org/2021.emnlp-main.243/.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 4582–4597. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.acl-long.353. URL https://aclanthology.org/2021.acl-long.353/.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/aca97732e30bcf1303bc22ac3924fd16-Abstract-Conference.html.

Luyang Liu, Jonas Pfeiffer, Jiaxing Wu, Jun Xie, and Arthur Szlam. Deliberation in latent space via differentiable cache augmentation. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 39261–39274, 2025. URL https://proceedings.mlr.press/v267/liu25bc.html.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Math-AI. AMC23. Hugging Face dataset, 2025. URL https://huggingface.co/datasets/math-ai/amc23.

Qwen Team. Qwen3-4B-Instruct-2507. Hugging Face model card, 2025. URL https://huggingface.co/Qwen/Qwen3-4B-Instruct-2507.

Qwen Team. Qwen3.5-4B. Hugging Face model card, 2026a. URL https://huggingface.co/Qwen/Qwen3.5-4B.

Qwen Team. Qwen3.5-9B. Hugging Face model card, 2026b. URL https://huggingface.co/Qwen/Qwen3.5-9B.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level Google-Proof Q&A benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024. doi: 10.1016/j.neucom.2023.127063.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, 2020. doi: 10.18653/v1/2020.emnlp-demos.6.

Yuhuai Wu, Markus N. Rabe, DeLesley Hutchins, and Christian Szegedy. Memorizing transformers. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=TrjbxzRcnf-.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=NG7sS51zVF.

Zeyu Xing, Xing Li, Huiling Zhen, Mingxuan Yuan, and Sinno Jialin Pan. Beyond speedup – utilizing KV cache for sampling and reasoning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/d147f24cac1b6cd88753ca830e462bdc-Abstract-Conference.html.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025a.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated Delta Networks: Improving Mamba2 with delta rule. In *The Thirteenth International Conference on Learning Representations*, 2025b. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/4904fad153f6434a7bcf04465d4be2cc-Abstract-Conference.html.

Jiayi Yao, Hanchen Li, Yuhan Liu, Siddhant Ray, Yihua Cheng, Qizheng Zhang, Kuntai Du, Shan Lu, and Junchen Jiang. CacheBlend: Fast large language model serving for RAG with cached knowledge fusion. In *Proceedings of the Twentieth European Conference on Computer Systems*, pp. 94–109. Association for Computing Machinery, 2025. doi: 10.1145/3689031.3696098.

Tianzhu Ye, Li Dong, Yuqing Xia, Yutao Sun, Yi Zhu, Gao Huang, and Furu Wei. Differential Transformer. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/00b67df24009747e8bbed4c2c6f9c825-Abstract-Conference.html.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, LingJun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Ru Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Yonghui Wu, and Mingxuan Wang. DAPO: An open-source LLM reinforcement learning system at scale. In *Advances in Neural Information Processing Systems*, volume 38, pp. 113222–113244. Curran Associates, Inc., 2025. doi: 10.52202/085713-3775. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/a4277440d50f1f15d2cb4c14f7e0c0d2-Abstract-Conference.html.

Yifan Zhang and Math-AI Team. American Invitational Mathematics Examination (AIME) 2025. Dataset, 2025. URL https://huggingface.co/datasets/math-ai/aime25.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark Barrett, and Ying Sheng. SGLang: Efficient execution of structured language model programs. In *Advances in Neural Information Processing Systems*, volume 37, pp. 62557–62583, 2024. doi: 10.52202/079017-2000. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/724be4472168f31ba1c9ac630f15dec8-Abstract-Conference.html.

A Historical Replay and Response Measurements

A.1 Replay Construction

The replay histories come from four-turn MATH-500 sessions generated by Qwen3-4B Instruct. Each session contains four distinct problems and four independently sampled paths. The first one, two, or three completed turns of a path form a source history. Each current problem is paired with histories whose source problems exclude it.

Two balanced designs vary coverage along the problem and history axes (Table 5). Within each design, the two data groups have disjoint current-problem sets and disjoint history-source pools. The history-coverage design crosses 64 current problems with 128 histories per group. The problem-coverage design expands the current-problem set to 128 and retains 32 histories. Figure 2 uses the three-turn problem-coverage condition, comprising 8,192 problem–history cells across the two groups.

Table 5: Balanced replay designs. Histories are sampled paths; sessions count distinct problem orders. Source problems count distinct problems in the first three turns.

Design	Group	Current problems	Histories	Sessions	Source problems
History coverage	1	64	128	33	81
History coverage	2	64	128	32	83
Problem coverage	1	128	32	25	67
Problem coverage	2	128	32	23	63

Across all three history depths, the history-coverage design contains 49,152 nonempty replays. The problem-coverage design reuses 12,288 cells and adds 12,288, giving 61,440 distinct nonempty replays in total. An empty-history reference is also computed for each of the 256 current problems.

All replays use a frozen backbone in BF16 with FlashAttention2 (Dao, 2024). Each history prefix is computed once and reused read-only across current problems. Replays use batches of 16 current problems, native causal masks, and position IDs following the realized conversation. The empty-history reference retains the system prompt and current-turn template. Across history depths, visible content, historical keys and values, and current-token positions vary together.

History length counts the complete tokenized prefix, including system and role-boundary positions. Median lengths at depths one, two, and three are 1,089.5, 2,196.5, and 3,016.5 tokens in group 1, and 849, 2,473.5, and 4,103 tokens in group 2. Three-turn lengths range from 1,259 to 19,136 tokens in group 1 and from 825 to 11,770 in group 2. Histories sharing any source problem are joined into a connected source block. Each problem-coverage group contains 17 such blocks, which define the source separation used in history pairing and resampling.

A.2 Attention and State Displacement

Let $\mathcal{T}_x$ contain the current user-body tokens, beginning after the user-role header and ending before the turn-closing delimiter. This span includes the shared answer instruction and the problem statement. We record outputs from all 36 transformer blocks, indexed from 0 to 35, with hidden width 2,560. The body-mean representation used in Section 2 is

$$h_\ell(x \mid H) = \frac{1}{|\mathcal{T}_x|} \sum_{t \in \mathcal{T}_x} h_{\ell,t}(x \mid H). \tag{11}$$

The corresponding state displacement is defined in Equation 1. Figure 2B reports the median of $\|d_\ell(x, H)\|_2$ over crossed cells. At layer 19, this median is approximately 17.76 in both groups; at layer 35 it is 62.07 and 61.97.

Let $\mathcal{S}_H^A$ denote earlier assistant-response body positions and n_q the number of query heads. For native attention probabilities $\alpha_{\ell,h,t,j}$, the attention mass on earlier assistant responses is

$$A_\ell^A(x, H) = \frac{1}{n_q |\mathcal{T}_x|} \sum_{h=1}^{n_q} \sum_{t \in \mathcal{T}_x} \sum_{j \in \mathcal{S}_H^A} \alpha_{\ell,h,t,j}. \tag{12}$$

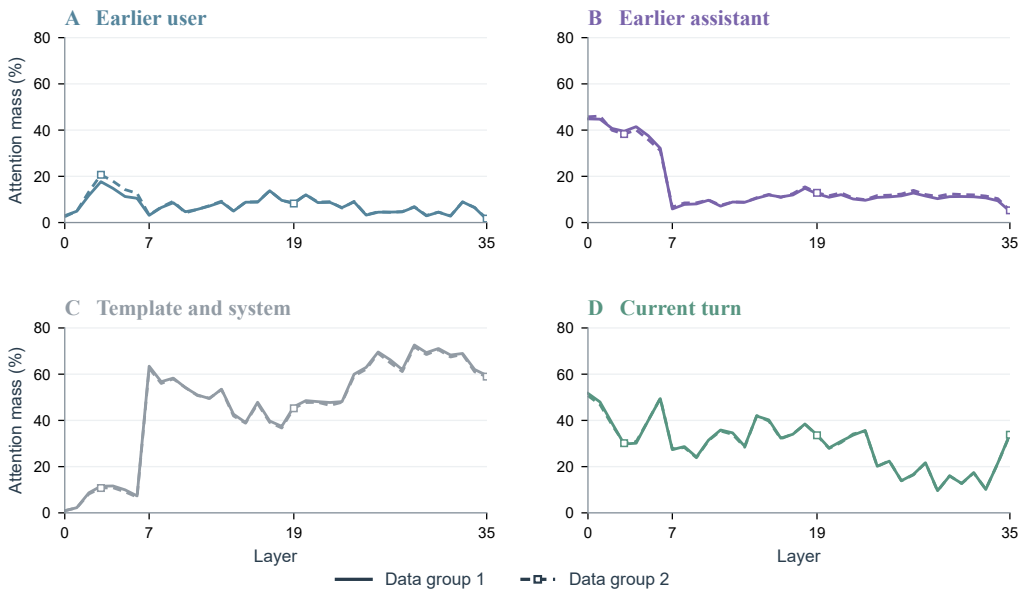

Figure 4: Native attention across source categories in the three-turn problem-coverage design. The four panels use common scales. Solid lines denote data group 1; dashed lines with hollow squares denote data group 2. Layer indices are zero-based.

Each head's probabilities are normalized over all causally visible positions. Earlier user bodies, earlier assistant bodies, earlier template and system positions, and current-turn positions form four disjoint source categories. Attention statistics are reconstructed in FP32. For each category, we take the median over current problems within a history, followed by the median over histories. Figure 4 shows their layerwise profiles. Because each category is aggregated separately, the plotted values need not sum to exactly 100% at a given layer.

A.3 Interaction Energy and Cross-History Geometry

For a fixed layer, history depth, and pooling view, let n and m denote the numbers of current problems and histories. The terms in Equation 2 are

$$\mu = \frac{1}{nm}\sum_{x,H} d(x,H), \qquad a(x) = \frac{1}{m}\sum_{H} d(x,H) - \mu, \tag{13}$$

$$b(H) = \frac{1}{n}\sum_{x} d(x,H) - \mu, \qquad e(x,H) = d(x,H) - \mu - a(x) - b(H). \tag{14}$$

Layer subscripts are omitted here. Under the balanced crossing, these components are orthogonal when summed over cells. In particular,

$$\sum_{x,H} \|d(x,H)\|_2^2 = nm\|\mu\|_2^2 + m\sum_{x}\|a(x)\|_2^2 + n\sum_{H}\|b(H)\|_2^2 + \sum_{x,H}\|e(x,H)\|_2^2. \tag{15}$$

The interaction energy fraction is the last term divided by total displacement energy. At layer 19 with three-turn histories, it is 0.69% in group 1 and 0.66% in group 2.

Neighborhoods are computed in the original 2,560-dimensional interaction-response space using Euclidean distance. Each group's 128 problems are split into four fixed folds, with 32 anchors and 96 candidate problems per fold. Every problem serves as an anchor once. Histories are sorted by realized length, and eight histories are selected at evenly spaced positions. Each is paired with the closest-length history from a different source block.



Figure 5: Cross-history neighborhood overlap at layer 19 under three-turn histories. Original coordinates use the full-dimensional interaction response; shared whitening uses projected coordinates with a common transform. Purple compares matched problem identities; gray shows the Question-identity shuffle control. Lines connect the three evaluated neighborhood sizes.

For an anchor x, let $\mathcal{N}_H^k(x)$ be its k nearest candidate problems under history H. The cross-history neighborhood overlap is

$$O_x(H, H'; k) = \frac{|\mathcal{N}_H^k(x) \cap \mathcal{N}_{H'}^k(x)|}{k}. \tag{16}$$

The Question-identity shuffle control permutes anchor identity in the second history within the same fold. Candidate identities, representations, and pairwise distances remain fixed. Overlap is averaged over anchors and folds within each history pair, then over the saved pairs.

Figure 5 gives the layer-19 results for three neighborhood sizes. A second coordinate view applies a fixed 128-dimensional Gaussian projection followed by shared whitening. Within each fold, candidate coordinates pooled across histories determine a common mean and covariance Σ. The whitening transform is $(\Sigma + \rho I)^{-1/2}$, with $\rho = 10^{-3}\,\mathrm{tr}(\Sigma)/128$.

At $k = 8$, raw state displacement yields overlaps of 73.83% and 69.82%. The additive reconstruction yields 100% in both groups: changing history adds a common translation to all problem representations. The interaction response retains 48.06% and 46.50% after those additive components are removed. Matched overlap also exceeds shuffled overlap across the larger neighborhoods and in the whitened coordinates.

A.4 Paired Answer Transitions

The MATH-500 evaluation includes all 500 problems with four sampled responses per problem at each session position, giving 2,000 responses per turn. Each later response is paired with Vanilla T1 by problem and sample identity. The transition analysis uses the scoring rules in Appendix C.3.

Table 6: Qwen3-4B Instruct on MATH-500: answer transitions relative to Vanilla T1. C and W denote correct and wrong answers. Net changes are in percentage points.

Turn	W to C	C to W	C to C	W to W	Net change
T2	37	47	1,853	63	−0.50
T3	40	59	1,841	60	−0.95
T4	35	60	1,840	65	−1.25

Both transition directions occur at every later turn (Table 6). Net changes equal the wrong-to-correct count minus the correct-to-wrong count, divided by 2,000 and multiplied by 100.

A.5 CONDITIONING ON THE FIRST HISTORY ANSWER

We group natural continuous sessions by the correctness of the actual T1 response in the same session and sampled trajectory. Native and STAIR share that response, so both conditions use the same grouping. At later turns, each follows its own generated history. The Vanilla reference answers the current problem independently and is matched by problem identity, sample index, and generation seed. Tables 7–9 report the resulting correct counts and conditional accuracies.

Table 1 pools T2–T4 across all three models on MATH-500, AIME 2025, and AMC23†. For each T1 group and condition, pooled accuracy is the sum of correct current answers divided by the sum of responses, multiplied by 100. Thus configurations contribute in proportion to their group sizes. The correct-T1 and wrong-T1 pools contain 18,879 and 1,425 later responses, respectively. GPQA-Diamond is reported separately in the tables below. These comparisons describe performance conditional on T1 correctness.

Table 7: Qwen3-4B Instruct: current-answer correct counts and conditional accuracy (%) by actual T1 correctness. Bold compares Native and STAIR within each group, including ties.

Turn	T1 answer	n	Vanilla	Native	STAIR
MATH-500					
T2	Correct	1,900	1,808 (95.16)	1,798 (94.63)	**1,806 (95.05)**
T2	Wrong	100	92 (92.00)	92 (92.00)	**94 (94.00)**
T3	Correct	1,900	1,805 (95.00)	1,787 (94.05)	**1,805 (95.00)**
T3	Wrong	100	95 (95.00)	**94 (94.00)**	**94 (94.00)**
T4	Correct	1,900	1,804 (94.95)	1,778 (93.58)	**1,784 (93.89)**
T4	Wrong	100	96 (96.00)	**97 (97.00)**	95 (95.00)
AIME 2025					
T2	Correct	57	25 (43.86)	**20 (35.09)**	19 (33.33)
T2	Wrong	63	32 (50.79)	24 (38.10)	**29 (46.03)**
T3	Correct	57	34 (59.65)	29 (50.88)	**30 (52.63)**
T3	Wrong	63	23 (36.51)	10 (15.87)	**13 (20.63)**
T4	Correct	57	25 (43.86)	16 (28.07)	**21 (36.84)**
T4	Wrong	63	32 (50.79)	**25 (39.68)**	**25 (39.68)**
AMC23†					
T2	Correct	126	116 (92.06)	118 (93.65)	**120 (95.24)**
T2	Wrong	10	10 (100.00)	**10 (100.00)**	9 (90.00)
T3	Correct	126	116 (92.06)	**117 (92.86)**	114 (90.48)
T3	Wrong	10	10 (100.00)	**10 (100.00)**	**10 (100.00)**
T4	Correct	126	116 (92.06)	116 (92.06)	**118 (93.65)**
T4	Wrong	10	10 (100.00)	**10 (100.00)**	**10 (100.00)**
GPQA-Diamond					
T2	Correct	464	279 (60.13)	**289 (62.28)**	278 (59.91)
T2	Wrong	328	185 (56.40)	**193 (58.84)**	185 (56.40)
T3	Correct	464	277 (59.70)	272 (58.62)	**276 (59.48)**
T3	Wrong	328	187 (57.01)	173 (52.74)	**190 (57.93)**
T4	Correct	464	274 (59.05)	270 (58.19)	**272 (58.62)**
T4	Wrong	328	190 (57.93)	181 (55.18)	**195 (59.45)**

† AMC23 excludes training-overlap problems, leaving 34 problems.

Table 8: Qwen3.5-4B: current-answer correct counts and conditional accuracy (%) by actual T1 correctness. Bold compares Native and STAIR within each group, including ties.

Turn	T1 answer	n	Vanilla	Native	STAIR
MATH-500					
T2	Correct	1,920	1,848 (96.25)	1,768 (92.08)	**1,813 (94.43)**
T2	Wrong	80	72 (90.00)	42 (52.50)	**59 (73.75)**
T3	Correct	1,920	1,844 (96.04)	1,864 (97.08)	**1,875 (97.66)**
T3	Wrong	80	76 (95.00)	67 (83.75)	**74 (92.50)**
T4	Correct	1,920	1,844 (96.04)	1,873 (97.55)	**1,878 (97.81)**
T4	Wrong	80	76 (95.00)	73 (91.25)	**77 (96.25)**
AIME 2025					
T2	Correct	73	47 (64.38)	29 (39.73)	**38 (52.05)**
T2	Wrong	47	26 (55.32)	7 (14.89)	**13 (27.66)**
T3	Correct	73	47 (64.38)	**42 (57.53)**	41 (56.16)
T3	Wrong	47	26 (55.32)	11 (23.40)	**19 (40.43)**
T4	Correct	73	40 (54.79)	40 (54.79)	**49 (67.12)**
T4	Wrong	47	33 (70.21)	17 (36.17)	**28 (59.57)**
AMC23†					
T2	Correct	104	76 (73.08)	62 (59.62)	**67 (64.42)**
T2	Wrong	32	28 (87.50)	9 (28.12)	**15 (46.88)**
T3	Correct	104	74 (71.15)	84 (80.77)	**90 (86.54)**
T3	Wrong	32	30 (93.75)	16 (50.00)	**25 (78.12)**
T4	Correct	104	79 (75.96)	86 (82.69)	**89 (85.58)**
T4	Wrong	32	25 (78.12)	**21 (65.62)**	18 (56.25)
GPQA-Diamond					
T2	Correct	504	316 (62.70)	387 (76.79)	**396 (78.57)**
T2	Wrong	288	188 (65.28)	209 (72.57)	**218 (75.69)**
T3	Correct	504	303 (60.12)	**369 (73.21)**	361 (71.63)
T3	Wrong	288	201 (69.79)	227 (78.82)	**231 (80.21)**
T4	Correct	504	339 (67.26)	**395 (78.37)**	382 (75.79)
T4	Wrong	288	165 (57.29)	201 (69.79)	**203 (70.49)**

† AMC23 excludes training-overlap problems, leaving 34 problems.

Table 9: Qwen3.5-9B: current-answer correct counts and conditional accuracy (%) by actual T1 correctness. Bold compares Native and STAIR within each group, including ties.

Turn	T1 answer	n	Vanilla	Native	STAIR
MATH-500					
T2	Correct	1,933	1,874 (96.95)	1,850 (95.71)	**1,857 (96.07)**
T2	Wrong	67	59 (88.06)	36 (53.73)	**46 (68.66)**
T3	Correct	1,933	1,867 (96.59)	**1,892 (97.88)**	1,884 (97.47)
T3	Wrong	67	66 (98.51)	54 (80.60)	**58 (86.57)**
T4	Correct	1,933	1,868 (96.64)	1,892 (97.88)	**1,900 (98.29)**
T4	Wrong	67	65 (97.01)	59 (88.06)	**62 (92.54)**
AIME 2025					
T2	Correct	74	52 (70.27)	47 (63.51)	**49 (66.22)**
T2	Wrong	46	22 (47.83)	10 (21.74)	**13 (28.26)**
T3	Correct	74	50 (67.57)	**41 (55.41)**	**41 (55.41)**
T3	Wrong	46	24 (52.17)	6 (13.04)	**16 (34.78)**
T4	Correct	74	41 (55.41)	**39 (52.70)**	38 (51.35)
T4	Wrong	46	33 (71.74)	13 (28.26)	**21 (45.65)**
AMC23†					
T2	Correct	106	80 (75.47)	79 (74.53)	**82 (77.36)**
T2	Wrong	30	26 (86.67)	12 (40.00)	**15 (50.00)**
T3	Correct	106	76 (71.70)	82 (77.36)	**87 (82.08)**
T3	Wrong	30	30 (100.00)	17 (56.67)	**22 (73.33)**
T4	Correct	106	82 (77.36)	**88 (83.02)**	86 (81.13)
T4	Wrong	30	24 (80.00)	**18 (60.00)**	17 (56.67)
GPQA-Diamond					
T2	Correct	545	376 (68.99)	**442 (81.10)**	409 (75.05)
T2	Wrong	247	169 (68.42)	**190 (76.92)**	162 (65.59)
T3	Correct	545	359 (65.87)	**432 (79.27)**	417 (76.51)
T3	Wrong	247	186 (75.30)	**201 (81.38)**	192 (77.73)
T4	Correct	545	381 (69.91)	447 (82.02)	**449 (82.39)**
T4	Wrong	247	164 (66.40)	**186 (75.30)**	175 (70.85)

† AMC23 excludes training-overlap problems, leaving 34 problems.

A.6 Historical Reads and Downstream MLP Responses

To examine how historical attention relates to later computation, we form the value-weighted read from earlier assistant positions:

$$r_\ell^A(x, H) = \frac{1}{|\mathcal{T}_x|} \sum_{t \in \mathcal{T}_x} W_{O,\ell} \operatorname{Concat}_h \left(\sum_{j \in \mathcal{S}_H^A} \alpha_{\ell,h,t,j} v_{\ell,g(h),j} \right), \tag{17}$$

where $g(h)$ maps query heads to their grouped key/value head and $W_{O,\ell}$ is the frozen output projection. Earlier-user and template/system reads use the corresponding source positions.

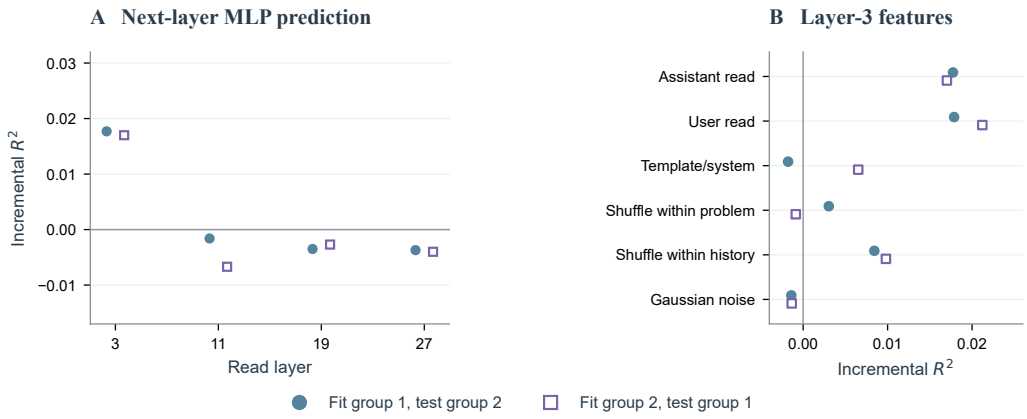

Figure 6: Historical reads and next-layer MLP prediction. **A:** Incremental test R^2 from adding the earlier-assistant read at four measured layers. **B:** Point estimates for equal-width feature additions at layer 3. Shuffling permutes assistant reads across histories within a problem, or across problems within a history, separately in fit and test groups. Colors and markers identify the two data-group exchange directions.

We test how much this read adds to a predictor of the next layer's MLP response. Let u_ℓ be the mean-pooled block input and z_ℓ the mean-pooled MLP output. The target is the full-dimensional difference $z_{\ell+1}(x \mid H) - z_{\ell+1}(x \mid \varnothing)$. The base feature contains 64-dimensional projections of $u_\ell(x \mid \varnothing)$ and its history-conditioned difference, log history length, and the three historical source-category attention masses:

$$f_0(x, H) = \left[Pu_\ell(x \mid \varnothing),\ P\Delta u_\ell(x, H),\ \log(1 + |H|),\ A_\ell^U,\ A_\ell^A,\ A_\ell^F \right]. \tag{18}$$

Here P is a fixed Gaussian projection, and F denotes template/system positions. Adding Pr_ℓ^A expands the feature from 132 to 196 dimensions. Pooled tensors are stored in FP32; regression uses FP64.

Multi-output ridge regression (Hoerl & Kennard, 1970) is fitted on one data group and evaluated on the other, with both exchange directions reported. Feature standardization uses only the fit group, and the ridge coefficient equals the number of fit cells. The incremental score is $\Delta R^2 = (\mathrm{SSE}_0 - \mathrm{SSE}_{\mathrm{read}})/\mathrm{TSS}$ on the held-out group.

The positive increment is concentrated at layer 3 (Figure 6A): ΔR^2 is 0.0177 when fitting group 1 and testing group 2, and 0.0170 in the reverse direction. At that layer, equal-width earlier-user reads carry comparable predictive information (Figure 6B). Shuffling assistant reads across histories within each problem reduces the increment to 0.00304 and -0.00087. These measurements describe an early association between historical reads and subsequent MLP responses.

B STAIR Implementation and Training

B.1 State Capture and Bank Accumulation

At each controlled attention layer, the historical K/V bank stores keys after projection and key normalization and values after value projection. Keys are captured before RoPE. The bank retains K/V heads in their original order. Under grouped-query attention, several query heads share one K/V head (Ainslie et al., 2023); each uses its corresponding stored head. Captured tensors are detached immediately and concatenated along the token dimension in turn order.

For Qwen3.5 models, the captured span begins at the start of the generated sequence and ends at the start of the last `</think>` marker. Ordinary history carries the final-answer text after that marker. If the marker is absent, the ordinary assistant message is empty and capture covers the

generated prefix that has actually undergone a forward pass. Instruct models retain the assistant-answer body in ordinary history and capture its processed states for the bank. A token sampled at the final decoding step has no captured key or value until it is subsequently processed by the model; this boundary also applies to responses ending at the length limit.

Training constructs banks during teacher-forced replay of the reorganized sessions. T1 uses the Native path; subsequent turns capture states from the student path. Capture is disabled during Native teacher evaluation. For each turn with a successor, the new entries are appended to the existing bank. Earlier entries keep the values captured when they were produced. Evaluation captures states during decoding. For reused T1 responses, we feed the saved tokens through the frozen backbone to construct the initial bank.

The ordinary decoding cache is local to a turn. At evaluation, both Qwen3 and Qwen3.5 re-prefill the full visible conversation at the start of each turn. The resulting ordinary cache serves that turn's autoregressive decoding. The auxiliary historical bank persists separately across turns, retaining states captured during their original generation even as the ordinary context is processed again.

B.2 Auxiliary Coordinates and Exact Reads

The concatenated bank receives auxiliary positions $0, \ldots, M_t - 1$. A current token uses $M_t + p$, with p measured in the full ordinary conversation input after removing left padding. The auxiliary branch applies the model's RoPE configuration to these positions, then reflects the positioned query as in Equation 5. Reflector normals are normalized in FP32 using a denominator floor of 10^{-8}.

Training and evaluation use temperature one and reference smoothing $\varepsilon = 10^{-6}$. With $c = \tilde{q}^R - \tilde{q}$, we center the logit change using the reference-weighted key mean:

$$\bar{k}_{\mathrm{ref}} = \sum_j \pi_j^{\mathrm{ref}} \tilde{k}_j, \qquad \pi_j^R = \mathrm{softmax}_j \left(\log \pi_j^{\mathrm{ref}} + \frac{c^\top (\tilde{k}_j - \bar{k}_{\mathrm{ref}})}{\sqrt{d_h}} \right). \tag{19}$$

For a fixed query, the centering term is constant across bank positions and cancels under softmax. This yields the reweighting in Equation 7, including its smoothed reference distribution.

B.3 Output Path and Prefill Scope

Let z concatenate the head-wise differential readouts at a controlled position. The auxiliary contribution to the attention output is

$$\Delta o = W_O z \quad \text{(Qwen3)}, \qquad \Delta o = W_O\big(g \odot z\big) \quad \text{(Qwen3.5)}, \tag{20}$$

where g is the native sigmoid output gate produced by the query projection. Its parameters and W_O remain frozen. The auxiliary projection uses the output weight alone; the native path retains the original output bias. The runtime adds Δo to the native self-attention output before the block performs residual addition and its MLP.

The control span starts at the last `<|im_start|>user` header and ends at the end of the prompt. It includes the user body, intervening template tokens, and assistant generation prefix. Qwen3.5 prompts also supply the opening `<think>` marker within this span. Earlier history positions and target-response positions have the auxiliary branch disabled (Figure 7(a)). At inference, the runtime disables the branch after prompt prefill and continues decoding from the resulting states.

B.4 Session Replay and Gradient Flow

Each model's response pool is generated with that same model through SGLang (Zheng et al., 2024) on problems from the training split of `BytedTsinghua-SIA/DAPO-Math-17k` (BytedTsinghua-SIA, 2025). Rollout settings appear in Appendix C.1. Each model uses 14,806 training problems and 128 validation problems. Responses are retained irrespective of correctness or completion status; correctness annotations carry no weight in the training objective. Some problems appear in multiple source records. Their responses remain grouped under the same problem identity.

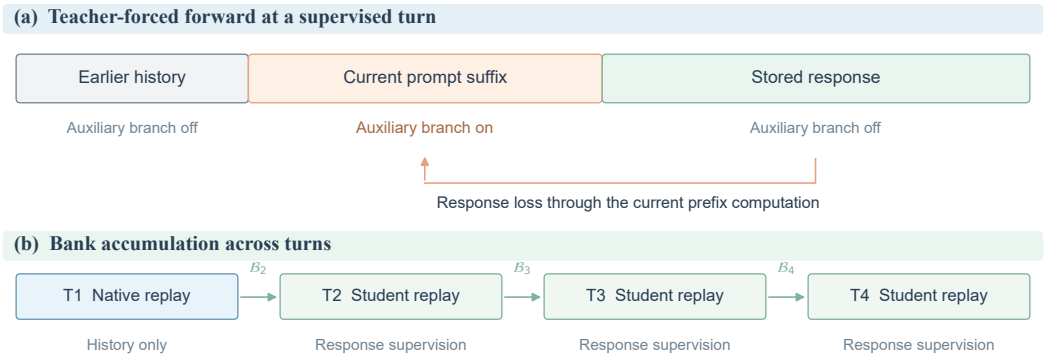

Figure 7: Training scope and bank accumulation. (a) The auxiliary branch acts on the current prompt suffix; response losses propagate through the current prefix computation. (b) T1 provides history, and T2–T4 receive response supervision. Each $\mathcal{B}_t$ contains detached states from all preceding turns. New entries come from the turn's replay and are appended for its successor.

Problems are grouped and matched to exclusion lists by normalized question text, removing fixed prompt prefixes and applying Unicode NFKC normalization, lowercasing, and removal of whitespace and selected LaTeX formatting. All three training pools use exclusion lists covering MATH-500, AIME 2025, and GPQA-Diamond. The 9B pool additionally excludes the 34-problem AMC23 evaluation subset. Six AMC23 problems retained in all three training pools are excluded from evaluation, as specified in Appendix C.3.

Validation is split at the problem level. Within each source record, the four recorded responses are randomly assigned one to each turn column. Columns are then shuffled independently, with swaps resolving repeated problem identities within a session. Each recorded response appears once, and each session contains four distinct problems.

Each training session contains four recorded problem–response pairs. T1 supplies the initial visible history and bank. For T2–T4, the student processes the prompt and recorded response in a causal teacher-forced forward pass. The control mask selects only the current-turn prompt suffix. Prediction targets cover the complete stored response, including reasoning, final answer, and any stored end markers. Incomplete responses contribute their existing tokens. The output at the final prompt position predicts the first response token; subsequent response positions predict the remaining targets.

The current-turn prefix computation retains its gradient graph. Response losses therefore reach the reflector normals through their dependence on the controlled prefix states. Captured bank tensors are detached, so later turns treat the accumulated bank as fixed input (Figure 7(b)).

The Native teacher independently processes the same ordinary history, prompt, and teacher-forced response prefix under disabled gradient recording. Where supported, earlier-history Native KV states can be reused as a detached prefix, ending before the current user message. Cross-turn Native KV reuse is disabled for the Qwen3.5 hybrid architecture, and its student forward runs without a KV cache. These choices preserve the current-turn gradient path while keeping the historical bank detached.

B.5 Supervision and Parameter Updates

The token objective in Equation 10 uses full-vocabulary $D_{\mathrm{KL}}(p_{\mathrm{Native}} \| p_{\mathrm{STAIR}})$ with coefficient one. The Native distribution is detached, while the student loss preserves gradients to the student hidden states.

For a supervised response r of length T_r, let w_r be its precomputed problem-balancing weight. An optimizer update over a set $\mathcal{U}$ of supervised responses uses

$$\mathcal{L}_{\mathcal{U}} = \frac{1}{|\mathcal{U}|} \sum_{r \in \mathcal{U}} w_r \left[\frac{1}{T_r} \sum_{a=1}^{T_r} \ell(y_{r,a}, s_{r,a}) \right]. \tag{21}$$

For Q training problems and S supervised responses in the full collection, a problem with C_q recorded candidates contributes $3C_q/4$ supervised responses. Each receives weight $w_r = (S/Q)/(3C_q/4)$. This gives every problem equal total weight while retaining all source rows. Update-level normalization uses the actual number of supervised responses across workers. T1 contributes history without a supervised response loss.

Only reflector normals receive optimizer updates. All three models control every query head at zero-based layer indices $[3, 11, 19]$. Each layer–head pair has its own normal, giving $\sum_{\ell \in \mathcal{L}} H_{q,\ell} d_{h,\ell} = 12{,}288$ trainable parameters per model (Table 10). Backbone parameters, the vocabulary head, the native output gate, and the attention output projection remain frozen.

In Qwen3.5, all three selected positions are native full-attention layers (Qwen Team, 2026a;b). STAIR reuses the keys and values produced at these layers. Gated DeltaNet layers (Yang et al., 2025b) retain their original computation without an auxiliary STAIR branch.

Table 10: Controller configuration. All query heads are controlled at layers $[3, 11, 19]$ (zero-based). Normals are separate for each layer and head.

Model	Query heads/layer	d_h	Init. std.	Parameters
Qwen3-4B Instruct	32	128	$1/\sqrt{128}$	12,288
Qwen3.5-4B	16	256	$1/16$	12,288
Qwen3.5-9B	16	256	$1/16$	12,288

At the start of training, normal coordinates are sampled independently as $n_{\ell,h,j} \sim \mathcal{N}(0, 1/d_h)$. The forward pass normalizes each vector as in Equation 5.

C Training and Evaluation Protocols

C.1 Training Settings

The backbones are `Qwen/Qwen3-4B-Instruct-2507`, `Qwen/Qwen3.5-4B`, and `Qwen/Qwen3.5-9B`. Training rollouts contain four responses per problem, with top-k 20, min-p 0, and repetition penalty 1. Instruct uses temperature 0.7, top-p 0.8, presence penalty 0, and a 16,384-token output limit. Both Qwen3.5 models use temperature 1.0, top-p 0.95, and presence penalty 1.5, with generation limits of 32,768 tokens for 4B and 131,072 for 9B.

Training problems follow the instruction "Please reason step by step, and put your final answer within `\boxed{}`.", followed by a blank line and the problem text. The model-specific rollouts have no system message. Instruct uses the non-thinking chat template; both Qwen3.5 models use the thinking template with an assistant prefix of `<think>` followed by a newline. Training replay uses the same thinking template and prefix.

STAIR uses AdamW (Loshchilov & Hutter, 2019) with learning rate 0.003, zero weight decay, and no scheduler or warmup. Training runs for one epoch in BF16 with FlashAttention2 (Dao, 2024). A full update contains eight four-turn sessions and 24 supervised responses at T2–T4; T1 supplies history. The objective combines CE with full-vocabulary $D_{\mathrm{KL}}(p_{\mathrm{Native}} \| p_{\mathrm{STAIR}})$ at coefficient and temperature one. Response-level token averaging and problem-balancing weights follow Appendix B.5.

The scheduled update counts are 1,851 for Qwen3-4B Instruct, 1,941 for Qwen3.5-4B, and 1,940 for Qwen3.5-9B. Checkpoints are selected at the final scheduled update, independently of validation or test performance. All models train 12,288 reflector parameters at zero-based layer indices $[3, 11, 19]$.

The LoRA baseline uses Qwen3-4B Instruct with rank 2, alpha 4, and dropout 0. Adapters cover `q_proj`, `k_proj`, `v_proj`, `o_proj`, `gate_proj`, `up_proj`, and `down_proj` in all 36 layers,

leaving embeddings, the vocabulary head, and biases unchanged. This configuration has 4,128,768 trainable parameters, 336 times the STAIR parameter count. It uses AdamW with learning rate 2×10^{-4} and zero weight decay, with 56 linear warmup steps followed by cosine decay to zero over the 1,851-step schedule. Training data, session construction, supervision, objective, and update budget match Instruct STAIR. Parameter count and optimization schedule differ; compute budgets have not been measured.

LoRA + STAIR initializes both modules afresh and trains them jointly, for a total of 4,141,056 parameters. Each module retains its configuration, learning rate, and schedule above. Data, session construction, objective, and the 1,851-step budget match the standalone Instruct runs. The Native teacher has both modules disabled. At evaluation, LoRA and LoRA + STAIR each generate their own T1 histories. In the joint model, LoRA acts at T1; STAIR begins at T2 once the historical bank is available.

C.2 Sampling and Matched Conditions

For a benchmark with N problems, we construct N four-turn sessions. Each turn column is a permutation of the benchmark: every problem occurs once at T1, once at T2, once at T3, and once at T4, across different sessions. The four problems within a session are distinct. The same problem can therefore supply history in one session and be evaluated at a later position in another. These assignments are fixed across models and conditions.

Each session is run along four sampled trajectories. Trajectory s at a later turn retains the earlier responses from that same trajectory. Seeds depend on benchmark, problem identity, and sample index, so the same problem–sample pair uses the same seed across positions and conditions. Vanilla evaluates every problem independently. Its saved responses also supply the shared T1 reference for Native and STAIR where reused.

Each problem has four sampled responses per turn. Native and STAIR use matched problems, reference answers, turn assignments, and sample seeds, with each condition continuing from its own generated history. Generation uses Transformers (Wolf et al., 2020) with FlashAttention2; Qwen3.5-9B runs in BF16. Table 11 gives the recorded decoding settings for all three models. All use $\texttt{min_p} = 0$, repetition penalty 1, and the corresponding official mode template.

Table 11: Recorded decoding settings. Output limits count newly generated tokens per problem.

Model	Temp.	Top-p	Top-k	Presence penalty	Output limit
Qwen3-4B Instruct	0.7	0.8	20	0	16,384
Qwen3.5-4B	1.0	0.95	20	1.5	81,920 / 32,768
Qwen3.5-9B	1.0	0.95	20	1.5	81,920 / 32,768

For both Qwen3.5 models, the 81,920-token evaluation limit applies to MATH-500 and AIME 2025; GPQA-Diamond and AMC23 use 32,768.

Evaluation uses the official chat templates without an additional system message, with thinking disabled for Instruct and enabled for Qwen3.5. Mathematical prompts read "Solve the following problem. Show your reasoning, and put the final answer inside `\boxed{}`.", followed on a new line by "Problem: `{problem}`". GPQA prompts read "Answer the following multiple-choice question. Reason carefully. Put your final answer, consisting of only the choice letter, inside `\boxed{}`, for example `\boxed{C}`.", followed by "Question:" and the problem on separate lines. The GPQA problem includes the question and its fixed A–D choice ordering.

C.3 Answer Scoring

For mathematical tasks, the scorer extracts the complete answer associated with the last `\boxed` occurrence in each response. Equivalence checks account for the requested answer format, including angle units, percentages, ordinals, and matrices. Integer-answer problems accept equivalent decimal or fractional representations and reject ambiguous multiple answers.

For GPQA-Diamond, each question has a fixed pseudorandom ordering of its four choices, shared across conditions. Scoring extracts the last boxed answer, removes supported formatting wrappers, and matches a single A–D choice letter to the reference label for that ordering, ignoring letter case and allowing surrounding parentheses.

Extraction failures are counted as incorrect. Responses reaching the output limit are scored on the available final answer, and the session continues to the next problem. Responses that exhaust the model's context capacity count as incorrect. Process failures without a saved response remain missing and are excluded from completed-result summaries.

When AMC23 was added to evaluation, normalized-text matching identified six problems present in all three training pools: original IDs $3, 15, 16, 19, 30, 32$ in the test split of `math-ai/amc23`. These problems remain in training and are excluded from both current questions and session histories for every evaluated condition. The resulting 34-problem subset is marked with † in tables and figures.

A turn-level score is reported only when all four responses are present for every benchmark problem. The per-turn denominators are 2,000 responses for MATH-500, 120 for AIME 2025, 136 for AMC23†, and 792 for GPQA-Diamond. A T2–T4 mean requires all three complete turns. Incomplete combinations remain unreported, so missing process outputs do not reduce the denominator of a reported score.

C.4 Shared-history Controls and Runtime Measurements

The T2 study uses Qwen3.5-4B and all 30 AIME 2025 problems, with four samples per problem. Its T1 trajectories were generated separately from those used in the continuous evaluation. Each of the 120 inputs contains a fixed T1 trajectory from the Native path. Exact-token replay through the frozen backbone constructs its historical bank using the capture boundaries in Appendix B.1. All conditions use the same visible T1 history, current problem, generation seed, and decoding budget. Only T2 answers are evaluated in this study.

Native and the Bank-free controller use ordinary conversation history without reading the auxiliary bank. STAIR uses its final controller and the captured T1 bank. The Random-reflector control replaces its normals with fixed random unit directions under three seeds. The Direct reflected-read control replaces $r^R - r^{\mathrm{ref}}$ with r^R while retaining the learned normals. Both modify the trained method at evaluation time.

The K–V pairing control starts from the same captured bank and permutes values across positions within each K/V head. Keys and auxiliary positions remain fixed. Query heads sharing a K/V head use the same permutation, which remains fixed throughout an answer. Both K–V pairing and Random-reflector cover three complete seeds (360 responses each); every other condition covers 120. All conditions use the scoring rules in Appendix C.3.

The Bank-free controller is trained separately with the same data, objective, update budget, controlled layers, and prefill positions as STAIR. Its 12,288 parameters are per-head, per-channel scales initialized to zero. During current-turn prefill, each scale multiplies the corresponding channel of the native attention output immediately before the frozen output projection. In Qwen3.5, the native output gate has already acted at this point. The controller reads no auxiliary bank.

The bank-prefix study uses the same 120 matched Qwen3.5-4B AIME 2025 inputs as the shared-T1 controls. It limits the readable bank to its first 2,048, 8,192, or 32,768 K/V entries in historical order; Full reads all entries. The visible conversation, stored K/V positions, and current-query virtual positions remain unchanged. A bank shorter than the budget is read in full. The median bank contains about 63.4K tokens. Table 12 reports results for all 30 problems under every condition.

Table 12: Bank-prefix accuracy on shared-T1 AIME 2025 at T2 (%). Each condition uses four matched sample seeds per problem. Truncated counts banks longer than the access budget.

Readable bank	Truncated	Avg@4	Pass@4
Native	–	21.67	56.67
STAIR 2K	120/120	35.00	73.33
STAIR 8K	114/120	43.33	**80.00**
STAIR 32K	81/120	39.17	70.00
STAIR Full	0/120	**48.33**	**80.00**

Runtime is measured on a single A800 80GB with Transformers, FlashAttention2, BF16, and batch size one. The 16 fixed Native MATH-500 T1 histories, selected without reference to answer correctness, each have a bank longer than 32K tokens and receive one warmup and three timed repetitions. We aggregate repetitions within each history, then report medians across histories (Table 13). The Full bank contains a median 80.7K tokens. Relative to Native, its paired median increases are 567 ms for T2 prefill, 0.81 s for the fixed workload, and 6.60 GiB for peak allocated memory. Exact-token replay to construct the bank takes a further median 7.29 s and is excluded from the table; full-answer free-generation time is not measured.

Table 13: Bank-size and fixed-workload costs across 16 histories. Transfer moves the bank to the GPU; total includes transfer, current-question prefill, and a fixed 256-token decode. Entries summarize each history before taking the median across histories.

Bank	K/V (MiB)	Transfer (ms)	Prefill (ms)	Decode (s)	Peak (GiB)	Total (s)
Native	0	≈ 0	70	10.08	8.03	10.15
2K	24	6	110	10.02	8.19	10.15
8K	96	24	157	10.06	8.70	10.23
32K	384	138	338	10.05	10.71	10.56
Full	≈ 946	330	628	9.98	14.61	10.95

D CONTINUOUS REASONING RESULTS

Tables 14–17 report Avg@4 and Pass@4 in percent by turn and as T2–T4 means. Bold marks the highest continuous-condition score at each later turn and in the mean, including ties. Dashes indicate turns not applicable to Vanilla.

D.1 QWEN3-4B INSTRUCT

Table 14: Qwen3-4B Instruct: results by turn (%). Mean averages T2–T4.

	T1		T2		T3		T4		Mean	
Condition	Avg@4	Pass@4	Avg@4	Pass@4	Avg@4	Pass@4	Avg@4	Pass@4	Avg@4	Pass@4
MATH-500										
Vanilla	95.00	97.20	–	–	–	–	–	–	–	–
Native	95.00	97.20	94.50	97.40	94.05	**97.60**	93.75	**97.80**	94.10	**97.60**
STAIR	95.00	97.20	**95.00**	**97.60**	**94.95**	**97.60**	**93.95**	97.40	**94.63**	97.53
AIME 2025										
Vanilla	47.50	66.67	–	–	–	–	–	–	–	–
Native	47.50	66.67	36.67	50.00	32.50	40.00	34.17	**53.33**	34.44	47.78
STAIR	47.50	66.67	**40.00**	**53.33**	**35.83**	**50.00**	**38.33**	**53.33**	**38.06**	**52.22**
AMC23†										
Vanilla	92.65	97.06	–	–	–	–	–	–	–	–
Native	92.65	97.06	94.12	97.06	**93.38**	**97.06**	92.65	94.12	**93.38**	96.08
STAIR	92.65	97.06	**94.85**	**100.00**	91.18	94.12	**94.12**	**97.06**	**93.38**	**97.06**
GPQA-Diamond										
Vanilla	58.59	78.79	–	–	–	–	–	–	–	–
Native	58.59	78.79	**60.86**	**78.28**	56.19	74.75	56.94	**77.78**	58.00	76.94
STAIR	58.59	78.79	58.46	75.76	**58.84**	**80.30**	**58.96**	77.27	**58.75**	**77.78**

† AMC23 excludes problems overlapping the training set, leaving 34 problems.

D.2 LORA COMPARISON AND JOINT TRAINING ON AIME 2025

Table 15: Qwen3-4B Instruct on AIME 2025: standalone and jointly trained adapters by turn (%). Mean averages T2–T4; bold marks the highest score in each column, including ties.

	T2		T3		T4		Mean	
Condition	Avg@4	Pass@4	Avg@4	Pass@4	Avg@4	Pass@4	Avg@4	Pass@4
Native	36.67	50.00	32.50	40.00	34.17	53.33	34.44	47.78
STAIR	40.00	53.33	35.83	50.00	38.33	53.33	38.06	52.22
LoRA	40.83	53.33	38.33	60.00	39.17	**63.33**	39.44	58.89
LoRA + STAIR	**44.17**	**63.33**	**45.00**	**70.00**	**40.83**	56.67	**43.33**	**63.33**

D.3 QWEN3.5-4B

Table 16: Qwen3.5-4B: results by turn (%). Mean averages T2–T4.

	T1		T2		T3		T4		Mean	
Condition	Avg@4	Pass@4	Avg@4	Pass@4	Avg@4	Pass@4	Avg@4	Pass@4	Avg@4	Pass@4
MATH-500										
Vanilla	96.00	98.60	–	–	–	–	–	–	–	–
Native	96.00	98.60	90.50	98.20	96.55	99.20	97.30	**99.00**	94.78	98.80
STAIR	96.00	98.60	**93.60**	**99.40**	**97.45**	**99.40**	**97.75**	**99.00**	**96.27**	**99.27**
AIME 2025										
Vanilla	60.83	80.00	–	–	–	–	–	–	–	–
Native	60.83	80.00	30.00	53.33	44.17	63.33	47.50	76.67	40.56	64.44
STAIR	60.83	80.00	**42.50**	**76.67**	**50.00**	**76.67**	**64.17**	**80.00**	**52.22**	**77.78**
AMC23†										
Vanilla	76.47	85.29	–	–	–	–	–	–	–	–
Native	76.47	85.29	52.21	73.53	73.53	**97.06**	**78.68**	**91.18**	68.14	87.25
STAIR	76.47	85.29	**60.29**	**82.35**	**84.56**	**97.06**	**78.68**	**91.18**	**74.51**	**90.20**
GPQA-Diamond										
Vanilla	63.64	75.76	–	–	–	–	–	–	–	–
Native	63.64	75.76	75.25	**89.39**	**75.25**	**88.89**	**75.25**	**89.90**	75.25	**89.39**
STAIR	63.64	75.76	**77.53**	**89.39**	74.75	86.87	73.86	84.34	**75.38**	86.87

† AMC23 excludes problems overlapping the training set, leaving 34 problems.

D.4 QWEN3.5-9B

Table 17: Qwen3.5-9B: results by turn (%). Mean averages T2–T4.

	T1		T2		T3		T4		Mean	
Condition	Avg@4	Pass@4	Avg@4	Pass@4	Avg@4	Pass@4	Avg@4	Pass@4	Avg@4	Pass@4
MATH-500										
Vanilla	96.65	98.20	–	–	–	–	–	–	–	–
Native	96.65	98.20	94.30	**98.80**	**97.30**	**99.60**	97.55	**99.20**	96.38	**99.20**
STAIR	96.65	98.20	**95.15**	**98.80**	97.10	99.40	**98.10**	**99.20**	**96.78**	99.13
AIME 2025										
Vanilla	61.67	73.33	–	–	–	–	–	–	–	–
Native	61.67	73.33	47.50	70.00	39.17	66.67	43.33	70.00	43.33	68.89
STAIR	61.67	73.33	**51.67**	**73.33**	**47.50**	**70.00**	**49.17**	**76.67**	**49.44**	**73.33**
AMC23†										
Vanilla	77.94	85.29	–	–	–	–	–	–	–	–
Native	77.94	85.29	66.91	**91.18**	72.79	91.18	**77.94**	91.18	72.55	91.18
STAIR	77.94	85.29	**71.32**	85.29	**80.15**	**94.12**	75.74	**97.06**	**75.74**	**92.16**
GPQA-Diamond										
Vanilla	68.81	80.30	–	–	–	–	–	–	–	–
Native	68.81	80.30	**79.80**	**90.91**	**79.92**	**89.39**	**79.92**	88.89	**79.88**	**89.73**
STAIR	68.81	80.30	72.10	85.86	76.89	88.38	78.79	**91.41**	75.93	88.55

† AMC23 excludes problems overlapping the training set, leaving 34 problems.