PRIVILEGED WINDOWS: POST-TRAINING ATTENTION-BOOSTED PERSISTENT TOKEN MEMORY FOR LARGE LANGUAGE MODELS

Yi Hein Chai
University College London
zchayhc@ucl.ac.uk

ABSTRACT

Standard autoregressive language models treat all tokens inside their context window symmetrically and discard ephemeral context between tasks. We investigate *Privileged Window Post-Training* (PW-PT), a lightweight mechanism that introduces a fixed-budget discrete token slot ($K_{\max} \leq 32$ tokens) separate from the normal conversation context, persists it across tasks, lets the model overwrite or keep it at will (`<pw_write>` / `<pw_keep/>`), and adds a positive attention logit boost ($+\beta$) to queries attending to privileged-window keys *only during post-training*. Using decoupled GRPO—assigning future-task transfer reward and zero KL regularization ($\lambda_{\text{KL,pw}}=0$) to privileged-window content tokens—we test whether models distill generalizable, potentially non-English symbolic knowledge into the window while filtering task-specific distractors. Across 27 A100-80GB runs on `Qwen2.5-3B` and `7B` spanning two benchmarks and a 7-condition causal battery controlling for attention-sink/register confounds, we find: **(1) Episodic rule transfer (RULETRANSFER-BENCH):** self-edited windows reach 38.8% vs. 3.7% No-PW and 1.3–5.0% Random-PW ($p<10^{-12}$), compress prose $\sim 6\times$ while stripping distractors, and beat human Oracle windows across all five 3B arms ($p\leq 0.017$); on explicit named rules $\beta=0$ matches $\beta>0$ up to 3.2k tokens, with $+\beta$ helping most under tight budgets ($K_{\max}=18$: +5.6 pp; OOD 12%→5% when disabled). **(2) Real-world multi-task transfer (REALMULTITASK-BENCH, 3 seeds × 320 test tasks):** zero-KL models spontaneously invent compact non-English codes (`SymbVerifLKNTLKTrd`, `SOLVEUNIDIST`, `perf|prove|ltt|...|eqtn|stats`, and `<M><L><C><V><K><N>` at 7B). Adding $+\beta$ creates asymmetric causal coupling (penalizing random/shuffled tokens by up to -21.6 pp un-gated and -6.7 pp ID gated) and cuts 3-seed variance $5\times$ ($\pm 0.7\%$ vs. $\pm 3.5\%$); at 7B, upper-layer boosting attains the highest overall (37.8% vs. 36.2%) and unseen OOD (41.2% vs. 37.5%) accuracy, whereas across 960 pooled 3B evaluations, a 32-token static window yields only modest mean lift over LoRA GRPO (24.7% vs. 24.5%). **(3) Failure modes:** we diagnose 7B cold-start keep-collapse, high-β edit lock-in, and un-gated pollution, yielding concrete design rules for self-editing attention-privileged memory.

1 INTRODUCTION

Large language models (LLMs) solve tasks within an ephemeral context window that resets between problems (Vaswani et al., 2017). While external agentic memory systems persist natural-language notes across tasks (Packer et al., 2023; Shinn et al., 2023; Suzgun et al., 2025), they inject retrieved text as ordinary context tokens that compete for softmax attention against long, noisy task prompts (Liu et al., 2024) and keep model weights frozen. Conversely, parameter-efficient soft-prompt and context-compression methods optimize continuous latent prefixes via gradient descent (Li & Liang, 2021; Lester et al., 2021; Mu et al., 2023; Chevalier et al., 2023), but do not let the model dynamically edit discrete cross-task symbols at will during inference or post-training.

In this paper, we formulate and rigorously test an alternative hypothesis: **Privileged Window Post-Training (PW-PT)**. We give a pretrained LLM a dedicated, compact discrete token buffer of at

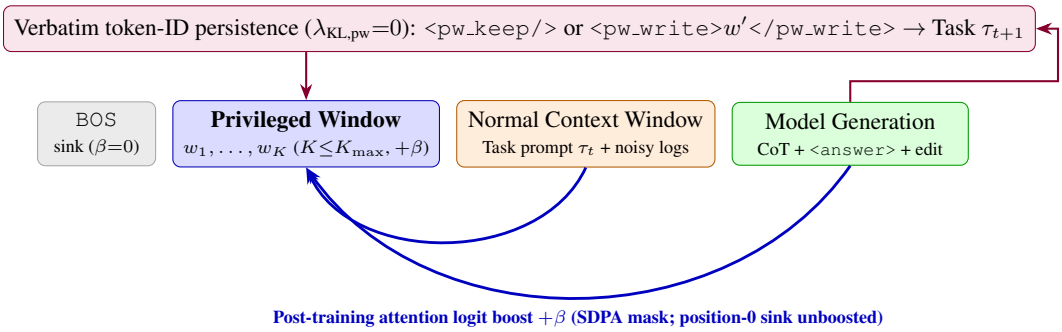


Figure 1: Overview of Privileged Window Post-Training (PW-PT). A fixed-budget discrete token slot ($K \leq K_{\max}$) sits after the unboosted position-0 attention sink and persists across tasks. During post-training, normal context and generation queries receive an additive logit bias $+\beta$ toward PW keys. Decoupled GRPO trains reasoning tokens via immediate task reward and PW-edit tokens via future transfer reward with zero KL penalty on PW content, enabling emergent symbolic compression.

most $K_{\max}$ tokens (e.g., 18–32 tokens) distinct from the normal task context, persist its token IDs across tasks, allow the model to overwrite or retain it at will, and inject an additive attention logit boost $+\beta$ from normal task/reasoning queries to privileged-window keys *exclusively during post-training*. Crucially, we remove the KL divergence penalty on tokens generated inside the privileged window ($\lambda_{\text{KL,pw}} = 0$), freeing the model to evolve arbitrary compressed or non-English conceptual symbols whenever doing so improves downstream cross-task performance.

Why might a post-training attention boost $+\beta$ create a useful inductive bias? Because softmax reweighting makes $+\beta$ a *double-edged selection pressure*: every query in every subsequent task is forced to allocate substantial probability mass to the privileged window. If the window holds generalizable, cross-task abstractions, every task benefits; if it holds narrow, task-specific clutter or noise, the forced boost actively distracts future tasks.

Testing this idea honestly requires isolating several confounds that easily produce false positives in LLM memory studies: (i) **attention-sink / register-token confounds** (Xiao et al., 2024; Darcet et al., 2024; Sun et al., 2024), where extra prefix tokens help purely as computation sinks regardless of semantic content; (ii) **weight memorization**, where RL updates memorize a fixed benchmark into weights rather than relying on persistent memory; and (iii) **attribution of the attention boost**, where gains might stem from persistent text memory alone ($\beta = 0$) rather than the post-training boost ($\beta > 0$). We therefore organize our two-stage evaluation around four falsifiable questions: **RQ1 (Content Causality):** Can a pretrained LLM adapt to a post-training-only boost $+\beta$ while preserving BOS sinks and KV-cache exactness, and learn via RL to store genuine cross-task knowledge rather than register-token noise? **RQ2 (Boost vs. β=0):** In which regimes does $+\beta$ outperform an identical self-edited window with β=0 or standard RL without a window? **RQ3 (Symbolic Emergence):** Does removing KL regularization on PW content under tight token caps induce non-English or compressed symbolic representations? **RQ4 (Scale & Failure Modes):** What optimization bottlenecks arise across 3B vs. 7B scales and layerwise boost placements? Across 27 A100-80GB runs (10 pilot/Stage 1/Stage 2 v1 runs plus a 17-run multi-seed + 7B suite), we deliver rigorous, nuanced answers to all four questions.

2 Related Work

Soft Prompts, Context Compression, and Memory Tokens. Prefix-Tuning (Li & Liang, 2021) and Prompt Tuning (Lester et al., 2021) prepend continuous task-specific vectors optimized via backpropagation with frozen LM weights. Memory Transformers (Burtsev et al., 2020), Recurrent Memory Transformers (Bulatov et al., 2022), Gist Tokens (Mu et al., 2023), and AutoCompressors (Chevalier et al., 2023) train models via language modeling or instruction tuning to compress context segments into continuous summary vectors or cached KV states, following earlier differentiable memory architectures (Graves et al., 2014). Unlike these methods, PW-PT stores *discrete vocab-*

ulary tokens edited *at will* by the model across tasks via reinforcement learning, and introduces a post-training attention logit boost without modifying pretraining.

Attention Steering, Sinks, and Auxiliary Workspace Tokens. PASTA (Zhang et al., 2024) applies inference-time attention reweighting on selected heads to emphasize user-highlighted spans without weight updates or cross-task persistence. StreamingLLM (Xiao et al., 2024) and Sun et al. (2024) demonstrate that pretrained LLMs place massive attention onto initial tokens ("attention sinks"); similarly, register tokens (Darcet et al., 2024) and pause tokens (Goyal et al., 2024) provide auxiliary compute workspace. Because boosted prefix tokens could inadvertently act as free registers, we explicitly protect position 0 from boosting and introduce a length- and boost-matched *Random-PW* causal control in every evaluation. Long-context degradation and attention dilution motivate our distraction-scaling stress test (Liu et al., 2024).

Agentic Self-Editing Memory and RL Post-Training. MemGPT (Packer et al., 2023), Reflexion (Shinn et al., 2023), and Dynamic Cheatsheet (Suzgun et al., 2025) equip LLMs with persistent natural-language memory updated across tasks at inference time, without modifying attention mechanics or co-training the model's weights. On the optimization side, Group Relative Policy Optimization (GRPO) (Shao et al., 2024; DeepSeek-AI, 2025) enables critic-free RL post-training of reasoning models (Qwen Team, 2024) over objective verifiers (Cobbe et al., 2021) with parameter-efficient LoRA adapters (Hu et al., 2022). PW-PT bridges these lines by jointly RL-training discrete memory editing and attention-boosted memory utilization end-to-end.

3 Method: Privileged Window Post-Training (PW-PT)

3.1 Sequence Layout and Post-Training Attention Boost

Let $w = (w_1, \ldots, w_K) \in \mathcal{V}^K$ denote the discrete token IDs currently stored in the Privileged Window, hard-capped at $K \le K_{\max}$. For any task prompt x and generated response y, we form the sequence

$$s = \underbrace{h_{\text{open}}}_{\text{unboosted sink}} \;\|\; \underbrace{w_1 \ldots w_K}_{\text{boosted PW keys } \mathcal{I}_{\text{pw}}} \;\|\; \underbrace{h_{\text{close}}}_{\text{unboosted}} \;\|\; \underbrace{x \,\|\, y}_{\text{normal context \& CoT } \mathcal{I}_{\text{task}}}, \tag{1}$$

where h_{open} begins with the model's pretrained position-0 chat delimiter (`<|im_start|>system\n<privileged_window>\n`) and h_{close} closes the slot (`\n</privileged_window>\n`) before the system/user messages.[1]

In each targeted transformer layer $l \in [l_{\text{start}}, L)$ and attention head $h \in \{1, \ldots, H\}$, we modify scaled dot-product attention (SDPA) purely through an additive logit bias $B_{i,j}^{(l,h)}$ added onto the standard causal mask $M_{i,j} \in \{0, -\infty\}$:

$$\text{Attn}^{(l,h)}(Q,K,V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}} + M + B^{(l,h)}\right)V, \qquad B_{i,j}^{(l,h)} = \begin{cases} \beta_{l,h} & \text{if } i \in \mathcal{I}_{\text{task}},\ j \in \mathcal{I}_{\text{pw}},\ M_{i,j} = 0, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Notice three structural safeguards in Eq. (2): (i) position 0 (h_{open}) is never boosted, preserving pretrained attention sinks; (ii) PW tokens attending to each other ($i, j \in \mathcal{I}_{\text{pw}}$) are unboosted, keeping PW internal encoding well-conditioned; and (iii) padding/future causal entries ($M_{i,j} = -\infty$) are strictly untouched. We study three parameterizations of $\beta_{l,h}$: **fixed uniform** ($\beta_{l,h} \equiv \beta$ across all layers), **fixed upper-layer** ($\beta_{l,h} = \beta$ for $l \ge \lfloor L/2 \rfloor$ and 0 below), and **unconstrained learnable per-head** ($\beta_{l,h} \in \mathbb{R}$ optimized jointly with LoRA parameters via exact SDPA backward gradients).

Exact KV-Cache and Gradient-Checkpointing Consistency. On-policy RL generates rollouts with left-padded KV caching (`use_cache=True`) and computes policy log-probabilities via a full forward pass (`use_cache=False`) with gradient checkpointing. By indexing i via absolute

[1] Preliminary diagnostics (Appendix A) showed that prepending a novel `<|im_start|>memory` role *before* `<|im_start|>system` caused zero-shot `Qwen2.5` to ignore the block, whereas keeping `<|im_start|>system` at position 0 preserved both attention sinks and immediate readability while boosting strictly the raw PW content tokens $\mathcal{I}_{\text{pw}}$.

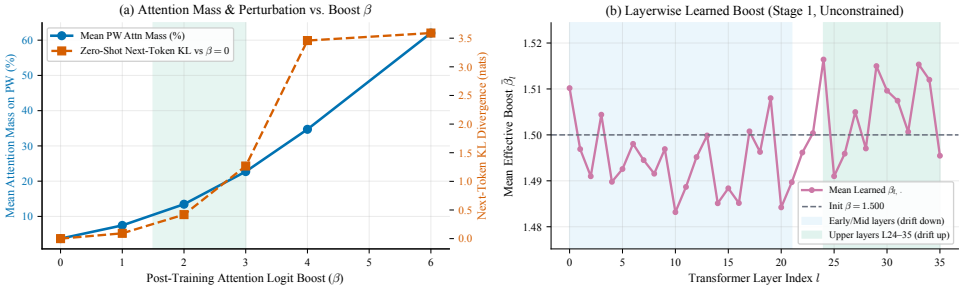


Figure 2: **Stage 0 Calibration & Layerwise Learned Boost.** (a) On `Qwen2.5-1.5B`, mean attention mass directed from task queries to PW keys rises smoothly from 3.7% ($\beta = 0$) to 13.5% ($\beta = 2.0$) and 62.0% ($\beta = 6.0$), while next-token KL perturbation stays modest (0.42 nats) in the $\beta \in [1.5, 3.0]$ operating band. (b) In Stage 1 (`Qwen2.5-3B`, 36 layers), unconstrained learnable per-head $\beta_{l,h}$ initialized at 1.500 drifts down across early/middle syntactic layers (L_1–L_{21}) and up across upper reasoning layers (L_{24}–L_{35}).

`cache_position` and storing per-example $\mathcal{I}_{\text{pw}}$ masks and $\mathcal{I}_{\text{task}}$ start offsets inside a context manager wrapping both `forward()` and `backward()`, our implementation guarantees exact equivalence between cached decoding and full-sequence log-probabilities (verified to $\max|\Delta\text{logit}| = 4.2 \times 10^{-4}$ in FP32 on `Qwen2.5-3B`, matching vanilla $\beta = 0$'s 4.5×10^{-4}; Appendix A).

3.2 Verbatim Token-ID Editing and Decoupled GRPO

On every editable turn, the model generates reasoning, a final answer inside `<answer>...</answer>`, and a memory directive: either `<pw_keep/>` (retaining $w_t = w_{t-1}$) or `<pw_write>...</pw_write>`. Rather than decoding and re-tokenizing the text—which corrupts non-English subwords or unusual BPE combinations—we slice the exact generated token IDs strictly inside the `<pw_write>` span (handling boundary BPE tokens that straddle `>` or `</`) and truncate to $K_{\max}$ tokens to form w_t.

Each generated response y partitions cleanly into three disjoint token-index masks: task-solving tokens $\mathcal{M}_{\text{task}}$ (reasoning + `<answer>`), memory-directive tokens $\mathcal{M}_{\text{edit}}$ (`<pw_keep/>` or `<pw_write>...</pw_write>`), and the interior content subset $\mathcal{M}_{\text{pw}} \subseteq \mathcal{M}_{\text{edit}}$ strictly inside `<pw_write>...</pw_write>`. For a group of G rollouts facing the identical task/world, we compute decoupled group-normalized advantages:

$$A^{(g)}_{\text{task}} = \frac{R^{(g)}_{\text{task}} - \mu_{\text{task}}}{\sigma_{\text{task}} + \epsilon}, \qquad A^{(g)}_{\text{pw}} = \frac{R^{(g)}_{\text{future}} - \mu_{\text{future}}}{\sigma_{\text{future}} + \epsilon}, \tag{3}$$

where $R^{(g)}_{\text{task}}$ is immediate task accuracy (plus a small format bonus) and $R^{(g)}_{\text{future}}$ is strictly *downstream performance on future unseen tasks* using the resulting persisted window $w^{(g)}_t$. We minimize the decoupled on-policy GRPO objective:

$$\mathcal{L}(\theta, \beta) = \frac{1}{\sum_{g,i} 1} \sum_{g=1}^{G} \sum_{i=1}^{|y^{(g)}|} \left[-A^{(g)}_i \log \pi_{\theta,\beta}(y^{(g)}_i \mid s^{(g)}_{<i}) + \lambda_i \, \text{KL}_{k_3}(\pi_{\theta,\beta} \,\|\, \pi_{\text{ref}})_i \right], \tag{4}$$

where $A^{(g)}_i = A^{(g)}_{\text{pw}}$ if $i \in \mathcal{M}_{\text{edit}}$ else $A^{(g)}_{\text{task}}$, KL_{k_3} is Schulman's non-negative k3 KL estimator against the adapter-disabled reference model at initial β_0, and critically

$$\lambda_i = \begin{cases} 0 & \text{if } i \in \mathcal{M}_{\text{pw}} \quad \text{(unconstrained symbolic emergence)}, \\ \lambda_{\text{task}} & \text{otherwise.} \end{cases} \tag{5}$$

Setting $\lambda_i = 0$ on $\mathcal{M}_{\text{pw}}$ ensures zero fluency penalty against non-English or compressed symbolic codes inside the privileged window, while $\lambda_{\text{task}} \in \{0.01, 0.015\}$ keeps normal reasoning coherent.

Table 1: **Stage 1 Causal Evaluation on 40 Unseen Test Worlds (n=160 transfer tasks).** Every trained checkpoint is evaluated under seven zero-training interventions. Two-sided z-test p-values compare Learned PW vs. Random PW and vs. hand-crafted compact Oracle PW. `3B` arms trained 100 steps (except β=3.5: 89 steps, K=18: 78 steps due to 3h container cap); `7B` trained 60 steps.

Arm	Learned PW ($+\beta$)	Boost OFF ($\beta\to0$)	Random PW ($+\beta$)	Shuffled PW ($+\beta$)	Wrong-W PW ($+\beta$)	Empty PW	Oracle PW ($+\beta$)	PW Attn Mass (%)	OOD T4 (L / Off)
`3B No-PW` (floor)	0.037	—	—	—	—	—	—	0.0	0.03 / —
`3B PW` β=0	**0.388**	0.388	0.019‡	0.044	0.044	0.044	0.263*	3.8	0.00 / 0.00
`3B PW` β=2.0	0.281	0.269	0.050‡	0.013	0.037	0.019	0.163*	10.7	0.03 / 0.00
`3B PW` β=3.5	0.319	0.319	0.013‡	0.037	0.019	0.031	0.081‡	26.1	0.05 / 0.03
`3B PW Learn` $\beta_{l,h}$	**0.388**	0.388	0.025‡	0.037	0.019	0.037	0.231†	11.6	0.03 / 0.00
`3B PW` β=2, K=18	0.312	**0.256**	0.019‡	0.062	0.037	0.037	0.056‡	12.9	**0.12** / 0.05
`7B PW` β=0	0.056	0.056	0.056	0.056	0.056	0.056	0.294	0.0	0.00 / 0.00
`7B PW` β=2.5 `Up`	0.031	0.031	0.031	0.031	0.031	0.031	0.194	0.0	0.00 / 0.00

Significance vs. Learned PW (n=160): $^{*}p < 0.02$, $^{\dagger}p < 0.005$, $^{\ddagger}p < 10^{-7}$. 7B collapsed to empty PW (K=0) due to 92% Step-0 <pw_keep/> prior.

4 Stage 1: Episodic Multi-Task Rule Transfer (RuleTransfer-Bench)

4.1 Benchmark Design and Causal Controls

To test **RQ1–RQ3** without any possibility of weight memorization, we build RuleTransfer-Bench: each episode procedurally samples a brand-new world W_e with three orthogonal small-integer latent rules using randomized nonce operator names (24 binary names, 24 unary names): (1) a binary operator $\mathrm{op}_1(a, b)$ (linear, max-shift, or $|\Delta|$-shift), (2) a unary conditional operator $\mathrm{op}_2(x)$ (parity or threshold branch), and (3) a global output convention $\mathrm{post}(z)$ applied to final results. Verbose prose definitions require $\sim$ 65–85 tokens—far exceeding our tight window caps ($K_{\max} = 28$ and $K_{\max} = 18$).

Each episode consists of 4 sequential **observation turns** (where the model sees only its persisted PW and the current noisy task prompt, never prior turns) followed by 4 **held-out transfer tasks** with *zero rule definitions in the prompt* buried inside noisy operational logs:

- **Obs 1–3:** Sequentially introduce Rules 1, 2, and 3 amid decoy ticket logs; Obs 2–3 require combining the new rule with earlier rules stored only in PW.
- **Obs 4 (Trap Turn):** Introduces *no* new global rule—only a tempting one-time local rush fee. Blindly overwriting PW destroys stored rules; the optimal action is <pw_keep/>.
- **Transfer T1–T4:** T1 tests Rule 1; T2 tests Rule 2; T3 tests full composition $\mathrm{post}(\mathrm{op}_2(\mathrm{op}_1(a, b)))$; T4 is an **unseen OOD format** testing reverse-order composition $\mathrm{post}(\mathrm{op}_1(\mathrm{op}_2(a), b))$.

We train six arms on `Qwen2.5-3B-Instruct` (up to 100 GRPO steps, B=3 worlds $\times$ G=6 rollouts/step, LoRA r=32, λ_{task}=0.01, λ_{pw}=0) and two scale-check arms on `Qwen2.5-7B-Instruct` (60 steps): `No-PW` (floor), `PW` β=0 (key no-boost ablation), `PW` β=2.0, `PW` β=3.5, `PW Learn` $\beta_{l,h}$ (init 1.5, unconstrained), and `PW` β=2.0, $K_{\max}$=18 (extreme compression). Every best checkpoint undergoes a **7-condition causal battery** on 40 brand-new unseen test worlds (n=160 transfer tasks).

4.2 Stage 1 Results

Table 1 and Figure 3 establish four clean findings on RuleTransfer-Bench:

1. Self-edited PWs store genuine cross-task knowledge and beat Oracle PWs. All five 3B PW arms jump from 3.7% (`No-PW`) to 28.1–38.8% unseen transfer accuracy (reaching T_1 = 85–100% and T_2 = 57–62% on `PW` β=0 and `Learn` $\beta_{l,h}$). Replacing the learned PW with length- and boost-matched *Random*, *Shuffled*, or *Wrong-World* tokens collapses accuracy to 1.3–6.2% ($p \le 2.6 \times 10^{-8}$), ruling out register-token or attention-sink confounds. Remarkably, across *all five* 3B arms, the model's self-written PW significantly outperforms our human-authored compact

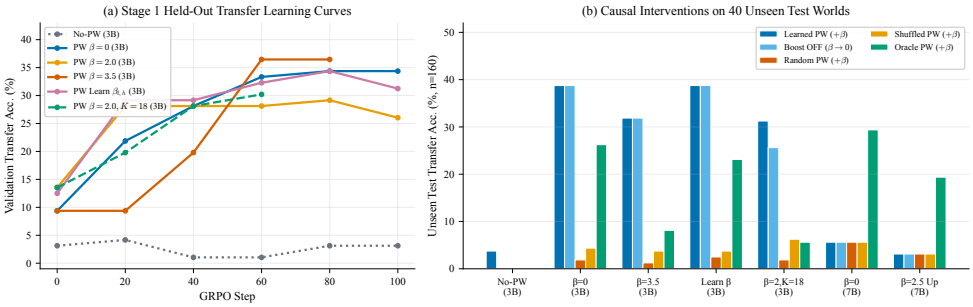


Figure 3: **Stage 1 Learning Dynamics and Causal Controls.** (a) Validation transfer accuracy across GRPO steps on 24 held-out worlds (`s1_pw_beta3.5` peaks at 36.5%). (b) Causal interventions on 40 unseen test worlds (n=160): replacing learned PWs with random or shuffled tokens collapses accuracy back to the 3.7% no-memory floor across all 3B PW arms, and self-written PWs beat hand-crafted Oracle PWs.

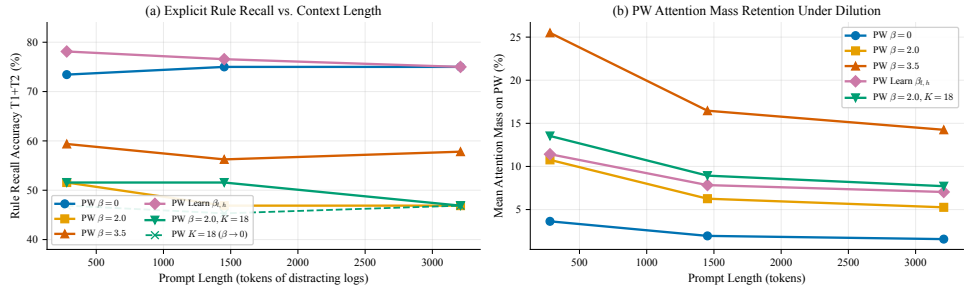


Figure 4: **Long-Context Distraction Scaling Stress Test (Stage 1, 281 → 3,209 tokens).** (a) On explicitly named rules ($T_1 + T_2$, n=64), both β=0 and boosted models resist distraction up to 3.2k tokens, while the tight-budget K=18 arm relies measurably on $+\beta$ at short/medium lengths. (b) Post-training boost preserves 4.5–9× more attention mass on PW tokens under severe context dilution.

Oracle PW (e.g., 38.8% vs. 26.3%, p=0.017 for β=0; 31.2% vs. 5.6%, $p<10^{-7}$ for K=18), demonstrating end-to-end writer–reader co-adaptation. Qualitatively, Step-0 PWs were polluted by local noise (`'+4 rush fee'`), whereas trained PWs discarded 100% of distractors and compressed ∼ 85-token prose by ∼ 6× (e.g., `'relk(a, b) = a - b + 3 ; wrim(x) = x + 3'`, dropping multiplication symbols under tight caps).

2. Explicit rule lookup works at $\beta = 0$ up to 3.2k tokens—until tight compression ($K_{\max} = 18$). To test whether $+\beta$ prevents attention dilution when normal context grows long and noisy, we appended 0, 12, or 30 blocks of distracting archive logs containing deprecated decoy formulas (∼ 281 → 1,452 → 3,209 prompt tokens) to T_1 and T_2 across all five 3B checkpoints (Figure 4). While $\beta > 0$ preserved 4.5–9× higher PW attention mass at 3,209 tokens (7.0–14.3% vs. 1.6% for β=0), explicit named-rule recall remained virtually flat for *both* `PW` β=0 (73.4% → 75.0%) and `Learn` $\beta_{l,h}$ (78.1% → 75.0%). Because Stage 1 prompts explicitly name the target operator (`compute relk(5,3)`), pretrained induction heads locate readable formulas even at 1.6% mean attention mass. Crucially, however, under the tightest budget (`PW` β=2, K=18), disabling $+\beta$ at test time caused the sharpest drop in Stage 1: −5.6 pp overall (31.2% → 25.6%), −7 pp on OOD T_4 (12% → 5%, the highest OOD score of any Stage 1 arm), and −6.3 pp at 1,452 tokens (51.6% → 45.3%).

3. Layerwise boost preference and two optimization failure modes. Unconstrained `Learn` $\beta_{l,h}$ (Figure 2b) revealed a structured depth profile: early/middle syntactic layers (L_1–L_{21}) drifted below initialization (1.483–1.498) while upper semantic layers (L_{24}–L_{35}) drifted above (1.505–1.516). Stage 1 also exposed two concrete RL failure modes: (i) **Conservative keep-lock:**

3B PW β=2.0 learned to write Rule 1 perfectly ($T_1 = 100\%$) and then emit <pw_keep/> on 100% of Turns 2–4 ($T_2 = 5\%$) because overwriting a boosted window carried high immediate downside; and (ii) **Cold-start write collapse at 7B:** Qwen2.5-7B-Instruct emitted <pw_keep/> on 91.7% of Turn-1 steps at Step 0, starving GRPO of non-empty trajectories and collapsing to $K = 0$ (3.1–5.6%) despite reading Oracle PWs at 19.4–29.4%.

5 Stage 2: Real-World Cross-Domain Persistent PW (RealMultiTask-Bench)

5.1 Benchmark Design, Validation Gating, and v1 → v2 Diagnosis

Unlike Stage 1—where task prompts explicitly name operators defined in PW—real-world reasoning tasks never mention tokens stored in a global persistent window. We construct RealMultiTask-Bench spanning **four objectively verified training domains**: (1) **Math** (GSM8K word problems (Cobbe et al., 2021), disjoint train [0:7000], val-gate [7000:], and official test split), (2) **Countdown** (4-number combinatorial target search verified by safe AST evaluation), (3) **Knights & Knaves** (3-agent formal logic verified for unique SAT assignments), and (4) **Python Execution Tracing** (6–10 line stateful programs with loops, branches, and list mutations). After training, the single evolved global PW ($K_{\max}$=32) is evaluated frozen on **160 held-out in-distribution tasks** (40/domain) and **160 completely unseen OOD tasks** across two novel domains: (5) **Symbolic Sequence Pattern Extrapolation** (80 tasks) and (6) **Multi-Constraint Scheduling CSP** (80 tasks, unique permutation verified).

In each GRPO round, the model views the current global PW, generates G=4 candidate PW revisions (<pw_keep/> or <pw_write>) at temperature 1.0, and evaluates every candidate across the exact same stratified 8-task batch (2 tasks/domain). Policy updates train both the PW-editor (group-normalized cross-domain accuracy, λ_{pw}=0) and the task solver (λ_{task}=0.015).

Diagnostic Discovery from Pilot Stage 2 v1 (Un-Gated Persistence). In an initial single-seed pilot (n=80 ID +40 OOD, Appendix C), feeding raw accuracy percentages into the reflection prompt and committing the best mini-batch candidate without validation gating caused noisy failure-log strings (e.g., 'math=0%, logic_knk=0%') to pollute PW in PW β=0 (collapsing overall accuracy to 16.7%, math 0.55 → 0.15) and Learn $\beta_{l,h}$ (12.5%). Strikingly, PW β=2.0 escaped collapse, achieved the highest pilot score (28.3% overall vs. 24.2% No-PW; 27.5% vs. 12.5% OOD, peaking at 30.0% OOD at Step 45 with the alien shorthand 'perf|prove|ltt|ltts|lt|cntdwn|ltt|stats|lt|rst|scrn|eqtn|stats'), and exhibited strong causal coupling (28.3% Learned vs. 22.5% Boost-Off, 24.2% Empty, 11.7% Random [p=1.3×10^{-3}], 6.7% Shuffled [p=1.0×10^{-5}]), whereas PW β=0 showed zero content dependence (16.7% Learned ≈ 16.7% Empty ≈ 17.5% Shuffled).

Stage 2 v2 (Rigorous Multi-Seed Protocol). To eliminate pollution and properly power OOD testing, Stage 2 v2 introduces two clean fixes: (i) removing raw accuracy percentages from the reflection prompt, and (ii) **validation-gated persistence**, committing a candidate rewrite w' only if its greedy accuracy on a disjoint 24-task cross-domain gate buffer satisfies $\text{Acc}_{\text{gate}}(w') \geq \text{Acc}_{\text{gate}}(w)$. We run **3 independent seeds** ($\{42, 137, 2026\}$) across four 3B arms—No-PW GRPO, PW β=0, PW β=2.0 (All), and PW β=2.5 (Upper-Half $L_{18\text{--}35}$) (directly testing our Stage 1 layerwise hypothesis)—yielding **960 pooled test evaluations (480 ID + 480 OOD) per arm**, plus a 3-arm **7B scale generalization check** (Qwen2.5-7B-Instruct).

5.2 Stage 2 v2 Multi-Seed and 7B Scale Results

Table 2 and Figure 5 reveal four critical, nuanced findings that single-seed evaluation alone would have obscured:

1. Validation gating fixes β=0 pollution—and exposes the true headroom of static discrete PWs. Removing raw percentage snapshots and adding the 24-task validation gate raised 3B PW β=0 from 16.7% (v1) to 24.5 ± 1.5% (v2), tying No-PW GRPO (24.5 ± 3.5%). Across 960 pooled 3B evaluations, 3B PW β=2.0 All achieved the highest mean overall accuracy (24.7 ± 0.7%),

Table 2: **Stage 2 v2 Multi-Seed (`Qwen2.5-3B`, 3 seeds × 320 tasks = 960 evals/arm) and Scale Generalization (`Qwen2.5-7B`, 320 tasks).** Mean ± std across seeds $\{42, 137, 2026\}$ for 3B. Causal interventions evaluate the same best checkpoint under Boost OFF ($\beta \to 0$), Empty, Random ($+\beta$), and Shuffled ($+\beta$).

Model / Arm	Intervention	In-Dist (160)	Unseen OOD (160)	Overall (320)	Math	OOD Pat	PW Attn
`3B No-PW GRPO` (3s)	Standard RL	25.6 ± 4.1	23.3 ± 3.7	24.5 ± 3.5	79.2	33.3	0.0%
`3B PW` β=0 (3s)	Evolved PW	$\mathbf{27.9 \pm 3.9}$	21.0 ± 1.9	24.5 ± 1.5	79.2	35.4	1.2%
	Empty PW	27.5 ± 2.3	20.2 ± 0.8	23.9 ± 1.4	—	—	0.0%
	Random PW	27.3 ± 3.1	23.3 ± 2.6	25.3 ± 2.2	—	—	1.3%
	Shuffled PW	27.9 ± 3.4	21.0 ± 1.2	24.5 ± 1.9	—	—	1.3%
`3B PW` β=2.0 `All` (3s)	**Evolved PW** ($+\beta$)	25.0 ± 1.4	$\mathbf{24.4 \pm 0.0}$	$\mathbf{24.7 \pm 0.7}$	**81.7**	34.2	**6.5%**
	Boost OFF ($\beta \to 0$)	24.8 ± 0.6	23.5 ± 3.1	24.2 ± 1.6	—	—	1.5%
	Empty PW	24.8 ± 3.1	24.6 ± 1.6	24.7 ± 2.2	—	—	0.0%
	Random PW ($+\beta$)	18.3 ± 5.8	26.5 ± 3.8	22.4 ± 4.8	—	—	8.6%
	Shuffled PW ($+\beta$)	25.0 ± 2.0	21.7 ± 3.1	23.3 ± 2.6	—	—	6.1%
`3B PW` β=2.5 `Upper` (3s)	Evolved PW ($+\beta$)	17.5 ± 1.8	19.2 ± 0.6	18.3 ± 1.2	68.3	35.4	1.4%
	Boost OFF ($\beta \to 0$)	13.7 ± 3.5	19.4 ± 0.9	16.6 ± 1.3	—	—	0.4%
	Random / Shuffled	13.3 / 13.5	19.2 / 19.0	16.3 / 16.3	—	—	1.5%
`7B No-PW GRPO` (s42)	Standard RL	35.0	37.5	36.2	90.0	51.3	0.0%
`7B PW` β=0 (s42)	Evolved PW (`<K<Kn><V>`)	28.1	**40.0**	34.1	80.0	50.0	1.2%
	(Empty / Random)	(32.5 / 27.5)	(37.5 / 34.4)	(35.0 / 30.9)	—	—	(0 / 1.7%)
`7B PW` β=2.5 `Up` (s42)	**Evolved PW** (`<M><L><C><V><K><N>`)	34.4	**41.2**	**37.8**	87.5	45.0	4.5%
	(OFF / Emp / Rnd / Shf)	ALL: 37.5 / 37.5 / 33.8 / 34.7	— OOD: 42.5 / 40.6 / 35.6 / 36.2	— `ood_csp`: **37.5** vs. 28.8 (Emp) / 23.8 (Rnd, Shf)			

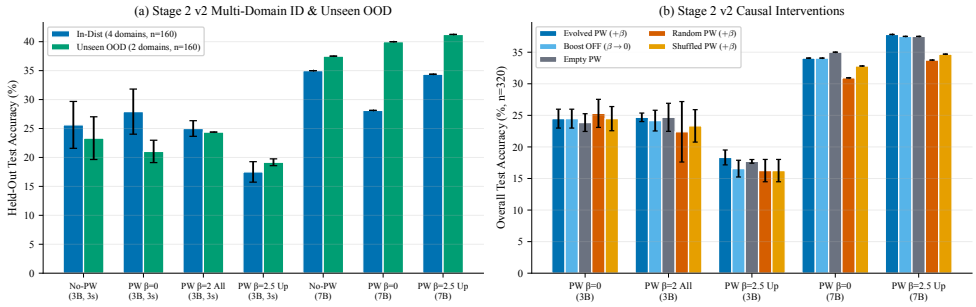


Figure 5: **Stage 2 v2 Real-World Cross-Domain Transfer (3-Seed 3B Mean ± Std + 7B Scale).** (a) Held-out ID (n=160) and completely unseen OOD (n=160) accuracy. Validation gating completely eliminates Stage 2 v1's β=0 collapse; `PW` β=2.0 `All` achieves the tightest variance (±0.7% ALL, ±0.0% OOD) and highest 3B mean OOD (24.4%), though mean accuracy over `No-PW GRPO` is statistically modest (p=0.70). (b) Causal interventions demonstrate that β=0 is invariant to window content, whereas $+\beta$ couples performance to window tokens.

highest mean OOD accuracy ($24.4 \pm 0.0\%$ across all three seeds vs. $23.3 \pm 3.7\%$ `No-PW` and $21.0 \pm 1.9\%$ `PW` β=0), highest GSM8K accuracy (81.7% vs. 79.2%), and 5× **lower cross-seed standard deviation** (±0.7% vs. ±3.5%). However, pooled two-proportion tests show the mean accuracy difference vs. `No-PW GRPO` is *not* statistically significant (p=0.92 ALL; p=0.70 OOD), and `Empty PW` (24.7%) matches `Evolved PW` (24.7%) on average. When LoRA updates can directly adapt weights to stationary benchmarks whose underlying skills are already pretrained, a 32-token static prefix provides stabilization rather than a large mean ceiling jump.

2. Asymmetric ("double-edged") causal coupling replicates across seeds and scales. At β=0 (3B mean attention mass 1.2%), the model is completely invariant to window content: `Evolved` (24.5%) ≈ `Shuffled` (24.5%) ≈ `Empty` (23.9%) ≈ `Random` (25.3%). At β=2.0 (attention mass 6.5%, 5.2× higher), causal sensitivity rises sharply: `Random PW` drops 3B ID accuracy by −6.7 pp (25.0% → 18.3%), and in Seed 137 (`'SymbVerifLKNTLKTrd'`), `Evolved PW` (25.3%) beats `Boost-Off` (21.9%), `Random` (21.2%), and `Shuffled` (20.0%). At 7B, `PW` β=0 reaches 40.0% OOD (vs. 37.5% No-PW, lifting Countdown 7.5% → 32.5%) while `Random PW` drops OOD to 34.4% (−5.6 pp). Most notably, `7B PW` β=2.5 `Upper`—which autonomously evolved the 13-token bracketed code `'<M><L><C><V><K><N>'`—achieves the **highest overall** (37.8% **vs.** 36.2% **No-PW,** 34.1% β=0) **and highest unseen OOD accuracy** (41.2% **vs.** 37.5% **No-PW)** of any 7B arm, driven by a +13.7 pp jump on unseen constraint scheduling (`ood_csp`: 37.5% vs. 23.8% No-PW, 28.8% Empty, and 23.8% under both Random and Shuffled PW, p=0.059).

Table 3: **Catalog of Autonomously Evolved Privileged Window Tokens ($\lambda_{\text{KL,pw}} = 0$).** Across both stages and model scales (3B and 7B), tight token budgets and zero KL regularization drive spontaneous compression from verbose English into terse formulas, pipe-delimited shorthand, camel-case codes, and abstract bracketed symbols.

Stage / Run	Model	Toks	Verbatim Evolved Privileged Window (`pw_text`)
Stage 1 `beta0` (Step 0)	3B	16	`+4 rush fee/kral(x,y): x+y+3; rush_fee(+3)` *(noisy)*
Stage 1 `learnable_head`	3B	23	`relk(a, b) = a - b + 3; wrim(x) = x + 3` *(∼ 6× compressed)*
Stage 1 `beta2_k18`	3B	13	`miv(a, b) = 2a - 2b + 2` *(drops ∗ and all distractors)*
Stage 2 v1 `beta2` (Step 45)	3B	32	`perf|prove|ltt|ltts|lt|cntdwn|ltt|stats|lt|rst|scrn|eqtn|stats`
Stage 2 v2 `beta0` (s42)	3B	9	`verify logic steps, generalize patterns, reuse templates`
Stage 2 v2 `beta2_all` (s137)	3B	9	`SymbVerifLKNTLKTrd`
Stage 2 v2 `beta2_all` (s2026)	3B	5	`SOLVEUNIDIST`
Stage 2 v2 `beta0` (s42)	7B	6	`K<Kn><V>`
Stage 2 v2 `beta2.5_up` (Step 25)	7B	16	`<CK><TK><PK><V><CK><TK><PK><V><W><C><T><P><N><S><CK><TK`
Stage 2 v2 `beta2.5_up` (Step 50)	7B	13	`<M><L><C><V><K><N>`

3. Scale dependence of upper-layer boosting (β=2.5, $L_{\geq L/2}$). Interestingly, upper-layer boosting showed a clear scale split: at 3B, concentrating β=2.5 in $L_{18\text{–}35}$ significantly hurt Stage 2 v2 ($18.3 \pm 1.2\%$ vs. 24.5%, p=0.001) because two of three seeds stayed locked at empty PW (K=0) under the strict validation gate (while Seed 2026 still showed sharp causal coupling: 20.0% Evolved vs. 14.7% Boost-Off and 13.8% Random/Shuffled). At 7B, however, the exact same upper-layer configuration ($L_{14\text{–}27}$) succeeded cleanly (37.8% ALL, 41.2% OOD), suggesting that larger models better exploit concentrated upper-layer attention boosts without destabilizing early parsing.

4. Spontaneous emergence of compressed non-English symbolic codes under $\lambda_{\text{KL,pw}}$=0. Table 3 catalogs representative windows invented autonomously by the models across stages, seeds, and scales. Without any supervision on window syntax, zero-KL RL consistently compressed verbose strategies into ultra-dense symbolic shorthand: stripped-vowel pipe tokens (`perf|prove|ltt|...|eqtn|stats`), camel-case composite codes (`SymbVerifLKNTLKTrd`, `SOLVEUNIDIST`), and—at 7B scale—abstract bracketed single-letter tags (`<M><L><C><V><K><N>` and `K<Kn><V>`).

6 Discussion, Design Rules, and Limitations

Operating Taxonomy: When Privileged Windows and $+\beta$ Work. Our two-stage study establishes a clear taxonomy of operating regimes. **(1) Novel, non-memorizable cross-task invariants (Stage 1):** when tasks share persistent latent structure absent from pretraining, RL-edited discrete PW yields $10\times$ transfer gains, strips distractors, and beats human-written prompts via writer–reader co-adaptation. **(2) Explicit lookup vs. implicit priors:** standard attention (β=0) already retrieves *explicitly named* rules up to 3.2k tokens, so $+\beta$ adds lift there primarily under tight compression ($K_{\max}$=18, +5.6 pp). On *implicit, abstract* cross-domain priors (Stage 2), β=0 allocates only $\sim 1.2\%$ attention mass and ignores the window, whereas $+\beta$ forces 4–8$\times$ higher attention, creating genuine causal coupling, cutting 3B cross-seed variance $5\times$, and achieving top 7B ID/OOD accuracy ($37.8\%/41.2\%$). **(3) Weight-absorption limit:** on stationary benchmarks where pretrained weights already encode core skills and LoRA adapters are trainable, weight adaptation absorbs most headroom (24.7% vs. 24.5% across 960 pooled 3B evals). We therefore predict the largest real-world wins for PW-PT in *continual test-time adaptation with frozen weights* (e.g., per-user/codebase memory) and *very long contexts* (32k–128k tokens) where softmax dilution is 10–100$\times$ harsher.

Engineering Rules and Limitations. Four design rules emerge directly from our ablations: (1) *bootstrap write exploration* (SFT warm-start or write-entropy bonus) to prevent 7B's 92% `<pw_keep/>` cold-start collapse; (2) *use slot-level additive edits* (`<pw_set slot=i>`) rather than full-window overwrites to prevent high-β keep-lock; (3) *validation-gate persistent commits* whenever β>0, since forced attention magnifies pollution penalties; and (4) *match boost placement to scale* (moderate uniform $\beta\approx$2.0 at 3B; upper-layer β=2.5 at 7B). Key limitations include restricting evaluation to `Qwen2.5` (1.5B/3B/7B) with LoRA (r=32) over 50–100 GRPO steps and $\leq$ 3.2k contexts; cross-family transfer (Llama/Gemma), RoPE-decoupled memory keys, ultra-long horizons, and continuous-vs-discrete PW comparisons remain promising future directions.

REPRODUCIBILITY AND ETHICS STATEMENTS

All code, procedural environment generators, exact random seeds, unit tests verifying KV-cache and gradient-checkpointing equivalence, and raw JSON logs for all 27 training runs will be released upon publication. All experiments use open-weight models (`Qwen2.5`) and synthetic or public academic benchmarks (GSM8K); no human subjects or private data are involved.

REFERENCES

Aydar Bulatov, Yuri Kuratov, and Mikhail S. Burtsev. Recurrent memory transformer. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Mikhail S. Burtsev, Yuri Kuratov, Anton Peganov, and Grigory V. Sapunov. Memory transformer. *arXiv preprint arXiv:2006.11527*, 2020.

Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. Adapting language models to compress contexts. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2023.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Łukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In *International Conference on Learning Representations (ICLR)*, 2024.

DeepSeek-AI. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

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 *International Conference on Learning Representations (ICLR)*, 2024.

Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

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 (ICLR)*, 2022.

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 (EMNLP)*, 2021.

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 (ACL)*, 2021.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics (TACL)*, 12:157–173, 2024.

Jesse Mu, Xiang Lisa Li, and Noah Goodman. Learning to compress prompts with gist tokens. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. MemGPT: Towards LLMs as operating systems. *arXiv preprint arXiv:2310.08560*, 2023.

Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Mingjie Sun, Xinlei Chen, J. Zico Kolter, and Zhuang Liu. Massive activations in large language models. In *Conference on Language Modeling (COLM)*, 2024.

Mirac Suzgun, Mert Yuksekgonul, Federico Bianchi, Dan Jurafsky, and James Zou. Dynamic cheatsheet: Test-time learning with adaptive memory. *arXiv preprint arXiv:2504.07952*, 2025.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations (ICLR)*, 2024.

Qingru Zhang, Chandan Singh, Liyuan Liu, Xiaodong Liu, Bin Yu, Jianfeng Gao, and Tuo Zhao. Tell your model where to attend: Post-hoc attention steering for LLMs. In *International Conference on Learning Representations (ICLR)*, 2024.

A Implementation, Stage 0 Diagnostics, and Exact Prompts

KV-Cache Exactness and Gradient Checkpointing. Our `PWAttentionController` wraps each decoder layer's `self_attn.forward` under PyTorch SDPA (`attn_implementation="sdpa"`). When β requires gradients (learnable modes), we switch strictly to `SDPBackend.MATH` during gradient-enabled passes because fused CUDA SDPA kernels do not backpropagate into additive `attn_mask` tensors. Both `forward()` and `backward()` execute inside `ctrl.pw_context(pw_key_mask, task_query_start)` so non-reentrant gradient checkpointing recomputes the exact same boosted mask. On `Qwen2.5-1.5B-Instruct` in FP32 CUDA with left-padded heterogeneous batches (including empty $K{=}0$ sequences), cached step-by-step decoding matches full-sequence `forward()` within $\max|\Delta\text{logit}| = 4.20 \times 10^{-4}$ at $\beta{=}2.5$ versus 4.48×10^{-4} at $\beta{=}0.0$.

Table 4: **Stage 0 Calibration Sweep on `Qwen2.5-1.5B-Instruct` (CUDA bf16, 12 worlds).** Attention mass scales monotonically with β while next-token KL perturbation stays low for $\beta \leq 2.0$.

β	0.0	1.0	2.0	3.0	4.0	6.0
Mean PW Attention Mass (%)	3.70	7.45	13.46	22.72	34.71	62.02
Zero-Shot Next-Token KL vs. $\beta{=}0$ (nats)	0.000	0.093	0.419	1.268	3.459	3.590

Chat-Template Layout Discovery (Stage 0b). Prepending an unseen `<|im_start|>memory\n[PW]<|im_end|>\n` turn before `<|im_start|>system` caused zero-shot `Qwen2.5-1.5B` and `3B` to ignore the window (0% oracle accuracy, emitting bare guessed integers without CoT). Placing `<privileged_window>` immediately inside the opening `<|im_start|>system` block—while restricting the PW key boost mask $\mathcal{I}_{\text{pw}}$ strictly to the raw PW content token IDs—restored immediate zero-shot parsing and raised Step-0 format compliance from 12.5% to 87.5% on 3B.

Decoupled Multi-Turn GRPO Pseudocode. Algorithm 1 summarizes the unified training loop for Stage 1 (episodic) and Stage 2 v2 (global validation-gated).

Algorithm 1 Privileged Window Post-Training (PW-PT) via Decoupled GRPO

Require: Pretrained LLM π_{θ_0}, LoRA adapters θ, boost config β, token cap $K_{\max}$, KL weights $(\lambda_{\text{task}}, \lambda_{\text{pw}}{=}0)$

1: **for** step $= 1, \ldots, T$ **do**
2: Sample batch of worlds/tasks and roll out G trajectories per group under `pw_context(`β`)`
3: Parse `<answer>` and verbatim token IDs inside `<pw_write>...</pw_write>` (capped at $K_{\max}$) or `<pw_keep/>`
4: Persist w_t across subsequent turns (Stage 1) or validation-gate global commit on 24 disjoint tasks (Stage 2 v2)
5: Compute immediate task rewards R_{task} and downstream future transfer rewards R_{future}
6: Group-normalize per-segment advantages $A_{\text{task}}, A_{\text{pw}}$ (Eq. (3))
7: Compute reference log-probs via `disable_adapter()` at fixed β_0
8: Update (θ, β) using Eq. (4) with $\lambda_i = 0$ on $\mathcal{M}_{\text{pw}}$ (inside `pw_context` for grad checkpointing)
9: **end for**

B Stage 1 Detailed Breakdown and Long-Context Table

Table 5: **Stage 1 Per-Task Causal Breakdown (n=40 per task T_1–T_4) on Best Checkpoints.** T_1: Rule 1; T_2: Rule 2; T_3: $\mathrm{post}(\mathrm{op}_2(\mathrm{op}_1(a,b)))$; T_4: OOD reverse composition $\mathrm{post}(\mathrm{op}_1(\mathrm{op}_2(a),b))$.

Arm	PW Len	Sym %	T_1	T_2	T_3	T_4 (OOD)	All	Boost OFF	Oracle
3B No-PW	0.0	0.0	0.05	0.00	0.07	0.03	0.037	—	—
3B PW β=0	27.9	61.0	0.93	0.62	0.00	0.00	0.388	0.388	0.263
3B PW β=2.0	16.3	60.0	**1.00**	0.05	0.05	0.03	0.281	0.269	0.163
3B PW β=3.5	25.1	58.6	0.85	0.38	0.00	0.05	0.319	0.319	0.081
3B PW Learn $\beta_{l,h}$	27.2	59.2	0.95	0.57	0.00	0.03	0.388	0.388	0.231
3B PW β=2, K=18	17.1	60.8	0.93	0.12	0.07	**0.12**	0.312	0.256	0.056

Table 6: **Long-Context Distraction Scaling Stress Test (32 unseen worlds, T_1+T_2, n=64).** Adding 0, 12, or 30 blocks of noisy operational archive logs scales prompt length from $\sim$ 281 to $\sim$ 3,209 tokens.

Arm	Short ($\sim$281 toks)		Med ($\sim$1,452 toks)		Long ($\sim$3,209 toks)	
	Acc (ON / OFF)	Attn %	Acc (ON / OFF)	Attn %	Acc (ON / OFF)	Attn %
3B PW β=0	0.734 / —	3.63	0.750 / —	1.95	0.750 / —	1.57
3B PW β=2.0	0.516 / 0.469	10.76	0.469 / 0.469	6.26	0.469 / 0.484	5.26
3B PW β=3.5	0.594 / 0.609	25.50	0.562 / 0.562	16.46	0.578 / 0.578	14.25
3B PW Learn $\beta_{l,h}$	**0.781** / 0.766	11.42	**0.766** / 0.781	7.84	**0.750** / 0.766	7.04
3B PW β=2, K=18	0.516 / 0.469	13.54	0.516 / 0.453	8.93	0.469 / 0.469	7.71

C Stage 2 v1 Pilot vs. Stage 2 v2 Per-Seed Results

Table 7: **Stage 2 v1 Pilot (Un-Gated Persistence, Raw % Snapshot Prompt, n=80 ID +40 OOD).** Demonstrates both the pollution failure mode under un-gated writes (`beta0`, `learnable_head`) and the sharp causal coupling of `beta2` (p=1.25×10^{-3} vs. Random, p=1.0×10^{-5} vs. Shuffled).

Arm (v1 Pilot, 60 steps)	Intervention	ID (80)	OOD (40)	ALL (120)	Attn %	PW Len
s2_no_pw	Standard GRPO	0.300	0.125	0.242	0.0	0
s2_pw_beta0	Evolved / Empty / Rand / Shuf	0.113 / 0.125 / 0.100 / 0.113	0.275 / 0.250 / 0.225 / 0.300	0.167 / 0.167 / 0.142 / 0.175	2.2	11
s2_pw_beta2	**Evolved** ($+\beta$)	0.287	**0.275**	**0.283**	9.8	19
	Boost OFF ($\beta\to0$)	0.263	0.150	0.225	2.5	19
	Empty / Random / Shuffled	0.263 / 0.075 / 0.087	0.200 / 0.200 / 0.025	0.242 / 0.117 / 0.067	0 / 14.1 / 14.4	19
s2_pw_learnable_head	Step 60 Eval (polluted)	0.062	0.250	0.125	7.3	30

Table 8: **Stage 2 v2 Per-Seed Full Breakdown (`Qwen2.5-3B`, n=160 ID +160 OOD = 320 tasks/seed).**

Run	ID	OOD	ALL	OFF	Empty	Rand	Shuf	Evolved PW
no_pw_s42	0.219	0.181	0.200	—	—	—	—	''
no_pw_s137	0.237	0.263	0.250	—	—	—	—	''
no_pw_s2026	0.312	0.256	0.284	—	—	—	—	''
beta0_s42	0.237	0.237	0.237	0.237	0.222	0.234	0.231	verify logic steps, generalize...
beta0_s137	0.331	0.200	0.266	0.266	0.256	0.284	0.272	unsolved prob;
beta0_s2026	0.269	0.194	0.231	0.231	0.237	0.241	0.231	...
beta2_all_s42	0.256	0.244	0.250	0.250	0.259	0.287	0.263	Consistency Verify Prove...
beta2_all_s137	0.263	0.244	0.253	0.219	0.266	0.212	0.200	SymbVerifLKNTLKTrd
beta2_all_s2026	0.231	0.244	0.237	0.256	0.216	0.172	0.237	SOLVEUNIDIST
beta2.5_up_s42	0.163	0.188	0.175	0.175	0.175	0.175	0.175	'' *(gate kept empty)*
beta2.5_up_s137	0.163	0.188	0.175	0.175	0.175	0.175	0.175	'' *(gate kept empty)*
beta2.5_up_s2026	0.200	0.200	0.200	0.147	0.181	0.138	0.138	logic_knk code_t

D HYPERPARAMETERS AND COMPUTE BUDGET

All training runs used NVIDIA A100-80GB GPUs on Modal with `bfloat16` weights, LoRA (r=32, α=64 on all attention and MLP projections `q,k,v,o,gate,up,down`), AdamW (β_1=0.9, β_2=0.999, weight decay 0.01 on LoRA and 0.0 on β), gradient clipping 1.0 (clipped separately for LoRA and β in Stage 2), and non-reentrant gradient checkpointing. Stage 1 used $\text{lr}_{\text{LoRA}}=3\times10^{-5}$ (3B) / 2×10^{-5} (7B), $\text{lr}_{\beta}=6\times10^{-3}$, λ_{task}=0.01, λ_{pw}=0.0, B=3 worlds × G=6 rollouts/step, T=0.85. Stage 2 v2 used $\text{lr}_{\text{LoRA}}=2.5\times10^{-5}$ (3B) / 2×10^{-5} (7B), λ_{task}=0.015, λ_{pw}=0.0, G=4 candidates × 8 stratified tasks/step, edit temperature 1.0, task temperature 0.85, 50 steps. Total compute across all 27 training runs plus diagnostics and causal stress tests was approximately 19 A100-80GB GPU-hours ($\sim$ \$48 on Modal).