Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Adaptive Inference-Time Reasoning for Large Language Models via Cross-Task Sparse Policy Transfer

Duo Xu

xu.duo3@northeastern.edu

Northeastern University

Zitan Chen

Carnegie Mellon University

Chenyi Huang

Johns Hopkins University

Short Report

Keywords: large language models, sparse guidance, policy transfer, inference cost, reasoning evaluation

Posted Date: November 10th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11669746/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: The authors declare no competing interests.

Adaptive Inference-Time Reasoning for Large Language Models via Cross-Task Sparse Policy Transfer

Duo Xu[1,*], Zitan Chen[2], Chenyi Huang[3]
[1] Northeastern University, Boston, 02115, USA
[2] Carnegie Mellon University, Pittsburgh, 15213, USA
[3] Johns Hopkins University, Baltimore, 21218, USA
[*]xu.duo3@northeastern.edu, zitanc@alumni.cmu.edu, chuan128@jh.edu

Abstract—Sparse guide queries need not yield proportional computation savings or better reasoning. We audit a frozen Qwen2.5-0.5B base/RL pair on 1,012 reference questions and an exploratory 128-question generation pilot. Aligned policy differences outperform a token-permuted control on 192 new questions per math/code domain. Local-harm AUROC reaches 0.826/0.871, but calibration, score ranking, ablations, and threshold scans expose selection failures. Adaptive gains of 0.00011/0.00039 nat/token lack a Benjamini–Hochberg-adjusted advantage over base; generation accuracy is 37.5% versus 38.3%. Against full-token RAST-form injection, calls fall 83.6% but guide input tokens only 25.5%, chiefly through avoided prompt prefill. This single-checkpoint audit distinguishes signal information, selection quality, and actual guide work.

Keywords—large language models, sparse guidance, policy transfer, inference cost, reasoning evaluation

I. Introduction and Related Work

Reasoning guidance must identify a decision worth changing and compute its correction. Chain-of-thought [1], self-consistency [2], and adaptive test-time allocation [3] use different inference budgets. Sparse token intervention alone does not establish low cost: an autoregressive guide must still process skipped prefix tokens before a later correction.

Policy adapters [4], proxy-tuning [5], and RAST [6] steer frozen models through auxiliary policies or RL-induced logit differences. Entropy [7] and disagreement [8] motivate selective intervention but do not establish cross-task benefit or expanded reasoning capacity [9].

We test whether target-only features identify useful corrections before querying a frozen RL guide. Immutable checkpoints, a token-permuted control, calibration/ablation diagnostics, and separate call/prefill/replay accounting make the audit traceable. Reference likelihood assesses local signals; generation remains exploratory. Unlike distribution-preserving speculative decoding [10], injection changes the sampled policy.

II. Adaptive Sparse Guidance

A. Checkpoint Coordinates and Policy Difference

The base/target is Qwen/Qwen2.5-0.5B [11], revision 060db6499f32faf8b98477b0a26969ef7d8b9987; the guide is jaygala24/Qwen2.5-0.5B-GRPO-math-reasoning [12], revision dff3c7eb54202cc44e95a0cf9e65c6b12d234ec6. Weight SHA-256 hashes and token-ID maps match the records. We reuse the provider's mathematical GRPO checkpoint without retraining. This same-size, single-checkpoint study does not independently replicate upstream RL or cross-scale transfer.

For frozen base p_t and guide r_t after prefix h_t, let $\Delta_t(v) = \log r_t(v) - \log p_t(v)$ and $m_t = (p_t + r_t)/2$. Disagreement is

$$D_t = \tfrac{1}{2} KL(p_t \parallel m_t) + \tfrac{1}{2} KL(r_t \parallel m_t). \quad (1)$$

A queried position uses the normalized correction

$$q_t(v) \propto p_t(v)^{1-\alpha_t} r_t(v)^{\alpha_t},$$
$$\alpha_t = min(0.5, \tfrac{0.05}{D_t + 10^{-8}}). \quad (2)$$

Base+RAST-full uses $q_t \propto p_t \exp(0.5\Delta_t)$ at every token, a same-size RAST-form comparator [6]. Zero Δ recovers base; coefficient 1 exactly recovers direct RL, so it adds no independent baseline. Sparse injection uses (2).

B. Pre-query Selection and Local Harm

Exact D_t requires guide inference, so selection uses eight target-only inputs: entropy, maximum probability, top-two log-probability margin, collision probability $\Sigma_v p_t(v)^2$, t/256, log(1+t), entropy × t/256, and squared margin. Source-training standardization, clipping to [−8,8], and degree-two expansion yield 44 features.

For reference token y_t, local gain and harm are

$$g_t = log\, q_t(y_t) - log\, p_t(y_t),$$
$$z_t = 1[g_t < -0.1]. \quad (3)$$

Three-output ridge regression (regularization 10) predicts D_t, g_t clipped to [−3,3], and z_t, using 135 coefficients/intercepts and 16 scaling statistics. Divergence/harm predictions are clipped to [0, ln 2]/[0,1]. These likelihood proxies do not measure final-answer correctness or causal error propagation.

$$s_t = \hat{g}_t + 0.1\hat{D}_t - 0.1\hat{z}_t(2 - min(t/256, 1)). \quad (4)$$

Hats denote predictions; the threshold is the source-validation 80th score percentile. Code trains the math selector, math trains the code selector, and mixed calibration serves BBH. Online selection uses no test references. Final-answer harm would require paired continuations; saved greedy trajectories do not establish it.

C. Query Caps and Replay Accounting

For generation cap L, sparse methods cap queries at ⌈0.2L⌉, all before t = ⌈0.5L⌉. Entropy uses its source-validation 80th-percentile threshold; random queries with probability 0.2 under identical caps. These ceilings do not equalize realized work. Each query caches skipped prefix tokens; for queried lengths $u_1 < \cdots < u_j$,

$$C_{guide} = u_1 + \Sigma_{i=2}^{j}(u_i - u_{i-1}) = u_j. \quad (5)$$

No-query cases cost zero guide work. Each query requests only the last-position logits and one vocabulary projection. We record calls/projections, prompt prefill, prefix replay, changed decisions, and synchronized time. Input-token counts are not attention FLOPs, memory traffic, or hardware-independent speed.

III. Experimental Protocol

A. Expanded Data and Fixed Generation Pilot

Table I covers GSM8K [13], MBPP [14], and two BBH tasks [15]. Math/code retain 96 training and 32 validation questions each; reference tests expand from 64 to 256 each. Both 250-question BBH sets are used without duplication. MBPP test item 217 duplicates training item 602 and is replaced deterministically. The original 512 records/traces are unchanged.

TABLE I. QUESTION COUNTS

Task	Train	Valid.	Ref. test	Generate
GSM8K	96	32	256	32
MBPP	96	32	256	32
BBH Boolean	0	0	250	32
BBH Deduction	0	0	250	32

Reference testing totals 1,012 questions; generation retains 32 per task. BBH references are short answers.

Selection uses seed 20261021 without outcome filtering. The archive records IDs, data, licenses (MIT/Apache 2.0/MIT), and hashes. Two math/code prompt examples are outside calibration/test sets. Prompts retain their last 512 tokens; reference and math/code generation caps are 256, BBH generation 16. Upstream training exposure and semantic duplicates remain possible. BF16 inference uses RTX 5060 Laptop GPU, PyTorch 2.9.1+cu128, and Transformers 4.57.1.

B. Controls and Statistical Units

Matched diagnostics select $\lceil 0.2T \rceil$ of T reference positions per question; full injection/direct RL use all positions. BBH's one-/three-token references preclude useful fractional masks, so sparse-position tests use math/code. A fixed vocabulary permutation of Δ (seed 20261028) preserves values but breaks alignment. Real/permuted half-strength injections are paired on 192 new questions per domain, separately from the 256-question comparison.

We test all seven score-component subsets, eight leave-one-input-out and eight single-input refits, plus three feature-group removals. A 4 × 5 scan crosses ridge {0.1,1,10,100} with source-validation quantiles {0.50,0.65,0.80,0.90,0.95}. Sequential masks obey 52-query/first-128-position limits on identical prefixes. The Pareto frontier is retrospective, not a deployed controller or measured latency frontier.

Questions are sampling units; token metrics and random seeds {7,19,43} average within question. Conditional 95% intervals use 5,000 question bootstraps; macro intervals resample tasks then questions. Calibration uses question-normalized weights and 2,000 cluster resamples. Fourteen paired sign-flip comparisons, two Delta-null tests, and four generation contrasts form separate Benjamini–Hochberg (BH) families [16], fixed before expanded analysis. Sign-flip tests assume pair exchangeability/sign symmetry; ablations/scans remain exploratory.

C. Evaluation Without Format-driven Credit

Generation compares base, direct RL, RAST-full, entropy, random, and adaptive. Math uses exact numeric matching after "####"; marker-free final-line extraction is a sensitivity. MBPP requires two hidden tests, adding the visible identifying test only in sensitivity; unrepaired programs use a timed restricted harness, which all 32 canonical solutions pass. BBH accepts one unique recognized answer, permits explanation, and rejects conflicts; first-match/whole-string rules are sensitivities. Fixed stopping/caps mean rescoring cannot recover truncated continuations.

IV. Results and Discussion

A. Does the Delta Signal Contain Useful Information

Table II separates signal from selection quality. Direct RL changes reference likelihood by −0.01455/0.00816 nat/token on math/code; half-strength full injection gives −0.00098/0.00678. Mathematical RL provenance thus does not guarantee mathematical likelihood gains. Adaptive gives only 0.00011/0.00039; on code, adaptive minus entropy is −0.00222 (BH q = 0.0007).

On 192 new questions per domain, aligned Δ exceeds the permuted control by 0.00404 nat/token on math (95% CI [0.00145, 0.00663], BH q = 0.0027) and 0.00997 on code ([0.00734, 0.01271], q = 0.0001). Alignment carries information, although aligned math injection itself has negative mean gain.

TABLE II. REFERENCE NLL REDUCTION

Method	Code → math	Math → code
Base	0.00000	0.00000
Direct RL	-0.01455	0.00816
Base+RAST-full	-0.00098	0.00678
Entropy, 20%	-0.00075	0.00261
Random, 20%	-0.00072	0.00118
Adaptive, 20%	0.00011	0.00039
Label oracle, 20%	0.02548	0.02258

Nat/token over all reference positions; positive is better, n = 256/domain. Columns identify selector source; all use the same RL checkpoint. Oracle uses test labels.

B. Discrimination Is Stronger Than Calibration

Local-harm AUROC is 0.826 on math (95% CI [0.816, 0.836]) and 0.871 on code ([0.858, 0.883]), despite poor calibration (Fig. 1). Predicted/observed harm rates are 2.54%/7.99% and 9.64%/2.89%; ten-bin calibration errors are 0.0546/0.0676. Code Brier error 0.0346 exceeds the prevalence-constant baseline 0.0280.

Code gain falls from 0.01367 to 0.00266 nat/token across increasing score quintiles; math is non-monotone. High scores can select safer positions while missing beneficial corrections. These reference-token outcomes do not establish calibrated final-answer harm.

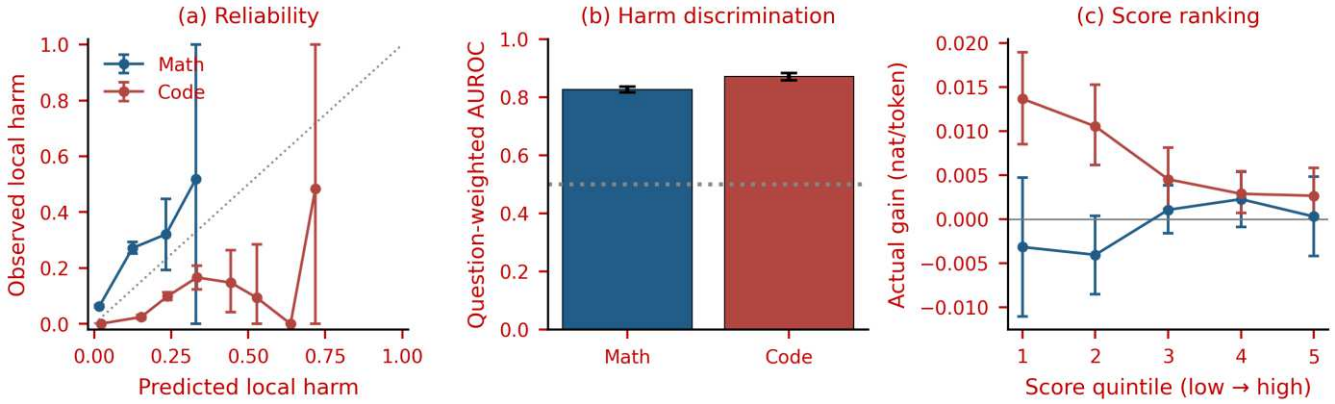


Fig. 1. Selector diagnostics, 256 questions/domain. Reliability uses ten fixed bins; score quintiles are label-free. Gains average within contributing questions, unlike Table II's all-position average. Error bars are question-cluster 95% intervals; high-probability bins have limited support.

C. Components Features and the Pareto Scan

Equal-position component ablations (Fig. 2) show conservative risk-only selection and little change from adding divergence. Position-only refitting gives 0.00242 nat/token on code versus 0.00039 for all inputs. Polynomial interactions reconstruct squared margin and entropy × position, so small deletion effects do not establish irrelevance; group removals address redundancy. Full refits/intervals are archived.

Higher thresholds reduce calls more than replay: one late query still processes its prompt/prefix. At ridge 10, code input work falls from 483.95 to 305.08 tokens across 50th–95th percentile thresholds, while gain drops from 0.00062 to −0.00003. Fixed-prefix scans imply neither generated trajectories nor timing; the original 80th-percentile operating point is retained.

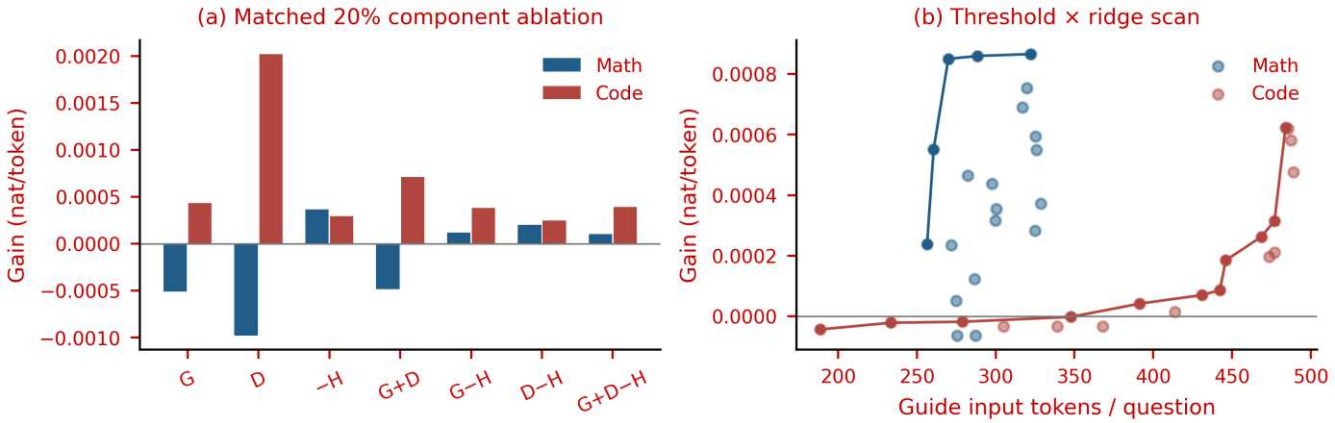


Fig. 2. (a) Seven combinations of gain G, divergence D, and negative horizon-weighted harm −H with equal 20% masks. (b) Fixed-prefix ridge/threshold scan; lines mark diagnostic nondominated points. Input work includes prefill/replay. Full refits/scans are archived; this is not an online accuracy/latency frontier.

D. Uncertainty and Exploratory Generation

Adaptive gain has 95% CI [−0.00081, 0.00100], BH q = 0.884 on math. Code's positive bootstrap CI [0.00004, 0.00082] coexists with sign-flip p = 0.0503, BH q = 0.106: multiplicity-adjusted benefit is unsupported. Different procedures need not agree. The hierarchical macro CI is [−0.00036, 0.00071]; two fixed tasks cannot characterize a broad task population.

At 80% design power and two-sided α = 0.05, detectable gains are 0.00130/0.00056 nat/token (math/code), or 0.00174/0.00076 with conservative 14-test planning. These are sensitivity estimates, not achieved power. Both 90% CIs fit exploratory ±0.001 equivalence bounds, but this revision-selected margin is not preregistered and does not prove zero effect.

TABLE III. EXPLORATORY GENERATION PILOT

Method	Primary	Exact format	Time (s)
Base	38.3	36.7	0.92
Direct RL	34.4	34.4	0.95
Base+RAST-full	35.9	35.9	1.91
Random	37.8	36.7	1.12
Entropy	35.2	33.6	1.14
Adaptive	37.5	35.9	1.04

Macro accuracy (%), 32 questions/task. Exact format changes only BBH scoring; all-three-test MBPP and marker-free math sensitivities leave scores unchanged. Shared-device time is descriptive.

The 128-question pilot yields adaptive/base accuracy 37.5%/38.3% (paired task-then-question 95% CI [−3.12, 0.00] percentage points; BH q = 1.00). Its sole discordance favors base. Whole-string BBH scoring rejects two correct explained answers per run, lowering both means equally. Parser

sensitivities do not change this exploratory gap; reasoning improvement is unproven.

E. What the Cost Reduction Actually Measures

TABLE IV. GUIDE WORK PER PILOT QUESTION

Guide measure	Full injection	Adaptive
Calls / projections	31.39	5.16
Prompt tokens	207.04	161.30
Replayed tokens	30.39	15.69
Total input tokens	237.43	176.98
No-query cases (%)	0.0	37.5

Means over 128 questions; each call includes one vocabulary projection. Input tokens count prefix processing, not FLOPs.

Calls fall 83.6% but guide input tokens only 25.5% (Table IV). Avoided prefill supplies 75.7% of token savings; 48/128 cases never query. Skipped tokens still require prefix processing before later queries, although intermediate vocabulary projections are omitted. Call savings therefore do not imply proportional total-computation savings.

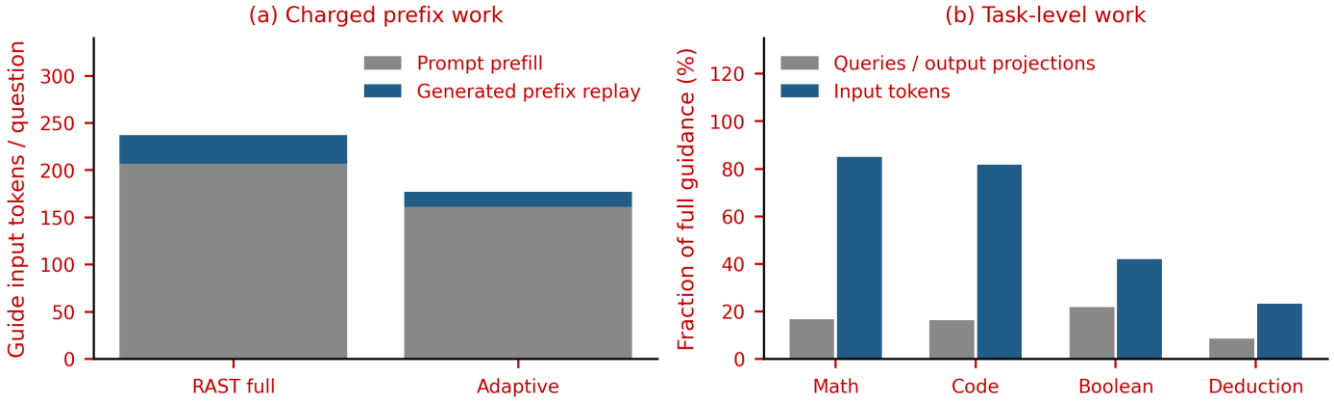


Fig. 3. Pilot guide work: (a) prompt prefill/prefix replay (exact means in Table IV); (b) task-specific adaptive/full ratios. Calls count vocabulary projections; input tokens are not FLOPs.

Math/code retain 85.0%/81.8% of full guide input work (Fig. 3); both checkpoints remain resident. Adaptive time is 1.04 s versus full injection's 1.91 s (−45.4%), but 13.0% above base. For 120 identical-output pairs, base/adaptive means are 0.833/0.857 s and the median adaptive/base ratio is 1.15. Output length, attention, contention, and replay batching confound timing attribution. The 660 adaptive queries change only 12 greedy decisions and add no correct answers, identifying an ineffective selector rather than useful acceleration over base.

V. Conclusion

Aligned policy differences contain information, but strong local-harm discrimination coexists with poor calibration and misranked benefits. The expanded selector comparison establishes no BH-adjusted advantage over base; the 37.5%/38.3% generation result remains exploratory.

Calls fall 83.6% versus 25.5% for input-token work, chiefly through avoided prefill. Reporting calls, prefill, replay, and outcomes jointly exposes this tradeoff. Evidence remains limited to one same-size checkpoint pair; selectors trained on paired continuation outcomes are the next test of improved complete reasoning.

References

[1] J. Wei et al., "Chain-of-thought prompting elicits reasoning in large language models," in Advances in Neural Information Processing Systems, vol. 35, 2022.

[2] X. Wang et al., "Self-consistency improves chain of thought reasoning in language models," in Proc. ICLR, 2023.

[3] C. Snell, J. Lee, K. Xu, and A. Kumar, "Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning," in Proc. ICLR, 2025.

[4] X. Lu et al., "Inference-time policy adapters (IPA): Tailoring extreme-scale LMs without fine-tuning," in Proc. EMNLP, 2023, pp. 6863–6883, doi: 10.18653/v1/2023.emnlp-main.424.

[5] A. Liu, X. Han, Y. Wang, Y. Tsvetkov, Y. Choi, and N. A. Smith, "Tuning language models by proxy," in Proc. COLM, 2024.

[6] S. Ouyang, X. Zhu, Z. Xiao, M. Jiang, Y. Meng, and J. Han, "RAST: Reasoning activation in LLMs via small-model transfer," in Advances in Neural Information Processing Systems, vol. 38, 2025, doi: 10.52202/085713-1663.

[7] S. Wang et al., "Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for LLM reasoning," in Advances in Neural Information Processing Systems, vol. 38, 2025, doi: 10.52202/085713-3850.

[8] C. Shen, L. Sheng, Y. Chen, X. Wang, and A. Zhang, "Reasoning can be restored by correcting a few decision tokens," in Proc. ICML, PMLR, vol. 306, 2026, pp. 111100–111124.

[9] Y. Yue et al., "Does reinforcement learning really incentivize reasoning capacity in LLMs beyond the base model?" in Advances in Neural Information Processing Systems, vol. 38, 2025.

[10] Y. Leviathan, M. Kalman, and Y. Matias, "Fast inference from transformers via speculative decoding," in Proc. ICML, PMLR, vol. 202, 2023, pp. 19274–19286.

[11] Qwen Team, "Qwen2.5-0.5B," model card and checkpoint, Hugging Face. Accessed: Oct. 21, 2026. [Online]. Available: https://huggingface.co/Qwen/Qwen2.5-0.5B

[12] jaygala24, "Qwen2.5-0.5B-GRPO-math-reasoning," model card and checkpoint, Hugging Face. Accessed: Oct. 21, 2026. [Online]. Available: https://huggingface.co/jaygala24/Qwen2.5-0.5B-GRPO-math-reasoning

[13] K. Cobbe et al., "Training verifiers to solve math word problems," arXiv:2110.14168, 2021.

[14] J. Austin et al., "Program synthesis with large language models," arXiv:2108.07732, 2021.

[15] M. Suzgun et al., "Challenging BIG-Bench tasks and whether chain-of-thought can solve them," in Findings of ACL, 2023, pp. 13003–13051, doi: 10.18653/v1/2023.findings-acl.824.

[16] Y. Benjamini and Y. Hochberg, "Controlling the false discovery rate: A practical and powerful approach to multiple testing," J. R. Stat. Soc. B, vol. 57, no. 1, pp. 289–300, 1995, doi: 10.1111/j.2517-6161.1995.tb02031.x.