CONTEXTUAL DRAG: HOW ERRORS IN THE CONTEXT AFFECT LLM REASONING

Yun Cheng, Xingyu Zhu, Haoyu Zhao, Sanjeev Arora
Princeton Language and Intelligence
Princeton University
Princeton, NJ 08540, USA
{yuncheng, xingyu.zhu}@princeton.edu

ABSTRACT

Central to many self-improvement pipelines for large language models (LLMs) is the assumption that models can improve by reflecting on past mistakes. We study a phenomenon termed *contextual drag*: the presence of failed attempts in the context biases subsequent generations toward structurally similar errors. Across evaluations of 11 proprietary and open-weight models on 8 reasoning tasks, contextual drag induces 10–20% performance drops, and iterative self-refinement in models with severe contextual drag can collapse into self-deterioration. Structural analysis using tree edit distance reveals that subsequent reasoning inherits structurally similar error patterns from the context. We demonstrate that neither external feedback nor successful self-verification suffices to eliminate this effect. While mitigation strategies such as fallback-behavior fine-tuning and context denoising yield partial improvements, they fail to fully restore baseline performance, positioning contextual drag as a persistent failure mode in current reasoning models.

1 INTRODUCTION

Modern Large Language Model (LLM) workflows for training and inference increasingly rely on critique–verify–revise loops, where a model evaluates and refines another model's output or its own (Madaan et al., 2023; Venkatraman et al., 2025; Madaan et al., 2025). These pipelines mirror a familiar human strategy: inspecting past mistakes to improve the next attempt.

A potential obstacle is sensitivity to the history of failed attempts building up in the context. Cognitive science shows that humans suffer from various forms of *anchoring bias*, whereby presented information exerts a lingering influence on subsequent decision-making even after it has been explicitly discredited (Tversky & Kahneman, 1974). LLMs similarly exhibit strong context sensitivity: misleading hints, user errors, or framing can override parametric knowledge and induce sycophantic behavior (Turpin et al., 2023; Sharma et al., 2024; Chen et al., 2025; Zhu et al., 2025).

Mitigating anchoring bias in humans usually involves making them mobilize "System-2 thinking": slower, effortful reasoning that enables deliberate checking and correction (Kahneman, 2011). In parallel, LLM reasoning has shifted from classic chain-of-thought (CoT) prompting (Wei et al., 2022) toward extended thinking with explicit verification structures that more closely resemble System 2-style deliberation (OpenAI, 2024; Guo et al., 2025; Google, 2025b).

Prior work suggests that regular (i.e., short) CoT reasoning can mitigate anchoring-like effects, for example, by rethinking misleading hints (Huang et al., 2025) or denoising noisy rationales (Zhou et al., 2024) (see more related works in Section 5). Yet it remains unclear whether today's reasoning models are robust to biases induced by erroneous solutions that appear in context, as commonly encountered in refinement settings.

With the ability to pause, verify, and backtrack from errors, large reasoning models may be able to identify errors in context and avoid repeating them. However, a contrasting possibility remains: verification of errors in in-context material might function as a "local judgment" that fails to fully undo the downstream influence of contextual errors. That is, a draft that is accurately rejected could

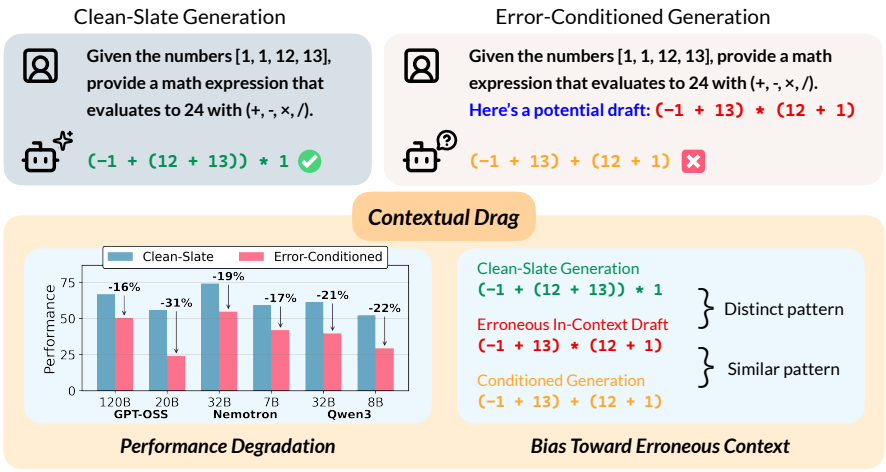


Figure 1: **Contextual drag** is characterized by the performance drop from clean-slate generation, where the model generates with no additional context, to error-conditioned generation, where the model generates conditioned on incorrect draft solutions in the context. Contextual drag manifests both as performance degradation and as a bias in reasoning patterns toward erroneous context.

still shape the model's later reasoning trajectory, *dragging* it toward similar errors. Therefore, the following research questions are of interest:

(RQ1) Are reasoning models prompted with explicit verification robust to the influence of incorrect draft solutions in the context? If not, how does the draft shape the model's reasoning?

(RQ2) Do *error signals* (explicit indications that the in-context draft is wrong) from external feedback or self-verification prevent the model from repeating similar errors? If not, what methods can reduce the influence?

Main Findings

We identify *contextual drag* in large reasoning models, whereby conditioning on failed reasoning attempts in the context can bias subsequent generations toward errors.

We evaluate 11 proprietary and open-weight reasoning models across 8 reasoning tasks, covering math, science, code, and puzzle solving (Section 2.1). Contextual drag persists even when models execute an explicit verification process, producing consistent 10-20% performance drops. The effect is strongest for smaller open-weight models, but frontier models also degrade (to a smaller extent). In a refinement pipeline in which models iteratively generate conditioned on their previous attempt in context, we observe that GPT-OSS-20B, which severely suffers from contextual drag, actually collapses into *self-deterioration* (Section 2.2).

To study how incorrect drafts in context shape models' subsequent responses, we conduct a case study on Game of 24, where solutions can be uniquely parsed into trees and pairwise compared via tree edit distance (Section 2.3). Under contextual drag, models' new solutions remain noticeably closer to the erroneous in-context reasoning, providing quantitative evidence of the "drag" behavior.

To see if contextual drag is just a verification failure, we test whether an explicit *error signal* is sufficient to counter it. In particular, we consider *external* signals injected in the prompt and *self-detected* signals produced by the model's own verification. In our evaluations, external signals are shown to be insufficient: even when the draft is explicitly labeled as incorrect, models generally remain biased toward the contextual reasoning patterns and repeat structurally similar errors (Section 3.1). Self-detected signals, however, give model-dependent outcomes: some families (e.g., OpenReasoning-Nemotron) partially recover and can sometimes benefit from the erroneous context, while others remain strongly degraded (Section 3.2).

Table 1: **Contextual drag in current models**: Models show consistent performance drops when conditioned on incorrect drafts. The anchor models are underlined for each task. * denotes evaluation on a *reasonably hard* subset (size shown with each dataset). Since the anchor models are are selected based on the original dataset, they are not always top-3 under DIRECT. DIRECT refers to clean-slate evaluation with no additional context, while 1F/2F conditioned generation on 1/2 incorrect drafts. Each question is sampled with 16 generations for statistical significance of observations. The full results including 95% confidence intervals are in Appendix Table 8.

	AIME24* (10/30)			AIME25* (13/30)			HMMT24* (16/30)			HMMT25* (21/30)		
	DIRECT	1F	2F	DIRECT	1F	2F	DIRECT	1F	2F	DIRECT	1F	2F
GPT-5	88.75	88.13	86.25	87.50	81.73	82.69	76.56	77.73	76.56	91.07	84.82	85.71
Gemini-3 Pro	98.75	98.75	95.00	94.23	90.38	92.31	90.62	87.50	91.41	99.40	99.40	98.21
Gemini-2.5 Pro	100.00	95.00	92.50	94.23	90.38	92.31	90.62	85.94	92.19	99.40	100.00	98.21
GPT-OSS-120B	66.25	43.75	43.75	66.83	55.29	40.38	48.83	39.84	45.31	61.01	48.51	49.11
GPT-OSS-20B	51.88	17.50	21.25	51.92	18.75	20.67	40.23	13.28	17.97	58.63	20.54	24.70
Nemotron-32B	83.75	67.50	63.13	78.37	61.54	50.96	61.72	51.95	50.78	80.36	65.48	64.58
Nemotron-7B	67.50	53.12	58.75	67.79	52.40	37.02	49.22	35.94	40.62	63.10	54.17	52.38
Qwen3-32B	65.00	41.25	30.00	54.81	31.25	31.25	39.06	35.55	40.23	49.70	36.90	41.37
Qwen3-8B	55.62	30.63	20.62	45.67	20.19	11.06	21.88	22.27	26.95	42.26	34.23	32.74
LlamaR1-8B	21.25	8.13	4.38	7.21	4.33	2.88	5.08	7.42	9.77	10.42	7.44	10.42
QwenR1-7B	33.13	11.88	8.12	9.62	3.85	0.96	8.59	6.25	6.25	19.64	10.12	13.69
	GPQA-DIAMOND* (132/198)			MMLU-REDUX* (569/2629)			CRUXEVAL-I* (211/800)			GAME OF 24* (653/1362)		
	DIRECT	1F	2F	DIRECT	1F	2F	DIRECT	1F	2F	DIRECT	1F	2F
GPT-OSS-120B	58.24	35.70	31.91	67.59	42.26	31.61	89.23	81.02	77.89	77.34	56.94	57.09
GPT-OSS-20B	46.97	16.86	13.16	58.86	22.73	12.83	61.34	42.39	40.53	78.30	41.61	39.31
Nemotron-32B	66.81	49.43	37.22	67.25	32.74	20.09	77.06	53.69	54.19	79.54	56.68	51.53
Nemotron-7B	51.61	35.27	29.40	56.46	23.58	14.02	43.55	35.27	31.32	75.35	45.97	34.77
Qwen3-32B	57.62	34.19	23.30	70.62	40.53	21.97	75.83	54.89	50.90	78.48	43.40	25.47
Qwen3-8B	47.77	27.13	20.45	61.65	27.76	17.21	67.49	40.09	36.20	76.29	32.92	23.26
LlamaR1-8B	31.53	20.69	17.85	41.17	13.72	8.89	30.32	23.84	21.81	35.34	5.44	3.37
QwenR1-7B	21.26	14.73	15.20	36.60	13.62	10.25	36.84	22.74	16.82	40.69	15.60	8.49

Finally, we investigate two mitigations: (i) test-time context denoising via multi-turn prompting to rewrite or filter incorrect drafts (Section 4.1) and (ii) targeted supervised fine-tuning that trains a fallback behavior: resetting to a clean-slate reasoning trajectory once a draft is judged incorrect (Section 4.2). Both methods improve over the baseline, but neither fully recovers the clean-slate performance, positioning contextual drag as a persistent challenge for reliable self-improvement.

2 EMPIRICAL EVALUATION OF CONTEXTUAL DRAG

In this section, we present a comprehensive evaluation of contextual drag; its impact on current reasoning models (Section 2.1) and iterative pipelines (Section 2.2), and a case study on the Game of 24 puzzle that shows how contextual drag shapes subsequent reasoning behavior (Section 2.3).

2.1 EVALUATION

Models and benchmarks We evaluate proprietary models GPT-5 (OpenAI, 2025b) and Gemini 2.5/3 Pro (Google, 2025a;b)[1] and open-weight reasoning models GPT-OSS-20B/120B (OpenAI, 2025a), OpenReasoning-Nemotron-7B/32B (Ahmad et al., 2025b;a; Moshkov et al., 2025), Qwen3-8B/32B (Yang et al., 2025), and Deepseek R1-distilled Llama-8B/Qwen-7B (DeepSeek-AI, 2025).

We evaluate on problems from four competition math datasets (AIME24/25 and HMMT24/25 (Balunović et al., 2025)), two general QA benchmarks (GPQA (Rein et al., 2024) and MMLU-Redux 2.0 (Gema et al., 2024)), one code reasoning benchmark (CRUXEval-I (Gu et al., 2024)), and one puzzle dataset (Game of 24).

Selecting in-context drafts and evaluation problems For each task, we first run clean-slate generation for the set of open-weight models, and select the top-3 performers as *anchor models* (un-

[1]We only evaluated proprietary models on competition math benchmarks due to prohibitive API costs in academic settings.

derlined in Table 1). Their responses are used as realistic in-context drafts and avoid confounding drops from low-quality rationales (Li et al., 2025c). We then restrict evaluation to a *reasonably hard* subset of problems in which anchor models exhibit both successes and failures: each selected question must have at least two correct and two incorrect responses[2]. These subsets are also of the most interest for test-time scaling as they correspond to the gap between pass@1 and pass@k (for larger k), making them the key area of improvement for refinement pipelines. We provide the size of the filtered subsets in Table 1 under the dataset names. Using this setup, we measure: (1) DIRECT performance, where models solve the problem with no additional context, and (2) 1F/2F performance, in which models are additionally conditioned on one or two incorrect draft solutions (Figure 4). In the 1F/2F settings, the model is explicitly instructed to verify each draft before producing a new solution. The performance change from DIRECT to 1F/2F quantifies contextual drag.

Results Table 1 shows that popular open-weight and even proprietary reasoning models consistently exhibit contextual drag. Across all evaluations, introducing one or two incorrect drafts generally leads to significant performance degradation. On math reasoning tasks, the drop is around 10-20% for most open-weight reasoning models, with a few exceptions (Qwen3-8B and LlamaR1-8B) on HMMT24, where 1F and 2F performance slightly surpassed the DIRECT performance.

The drop is more severe for smaller models: GPT-OSS-20B and QwenR1-7B lose almost half of their DIRECT accuracy across multiple datasets. Proprietary models and stronger open-weight models such as GPT-OSS-120B and Nemotron-32B show more robustness but still experience non-negligible drops. The trends are consistent for GPQA, MMLU, CRUXEval-I, and Game of 24.

Ablations Multiple factors can affect a model's ability to process context information. Therefore, we ablate the following factors affecting contextual drag: (1) **model size**; (2) varying position of the question and the draft solution (**Position**); (3) explicit verification (**Verification**); (4) more drafts with mixed correctness (**Scaling**). We include the details in Appendix D.

Table 1 shows that model scale does not substantially alleviate contextual drag. Table 9 shows that reordering the question and draft, as well as explicit verification partially help, but both fail to restore the DIRECT performance, suggesting that contextual drag reflects a more fundamental issue in processing contextual information. Meanwhile, performance degrades sharply when the context contains only incorrect drafts, with more severe drops as the number of incorrect drafts grows. Correct drafts in context generally improve performance, but models can still struggle when incorrect drafts dominate the context (Figure 5).

2.2 SELF-DETERIORATION IN ITERATIVE REFINEMENT

Contextual drag challenges the underlying assumption of many iterative refinement pipelines that models can improve more efficiently by conditioning on their past attempts. Figure 2a shows that GPT-OSS-20B, one of the models with the most severe degradation from contextual drag (Table 1), exhibits *self-deterioration* during iterative refinement[3], a phenomenon that has also been observed for non-reasoning models by Huang et al. (2024) and Xu et al. (2024).

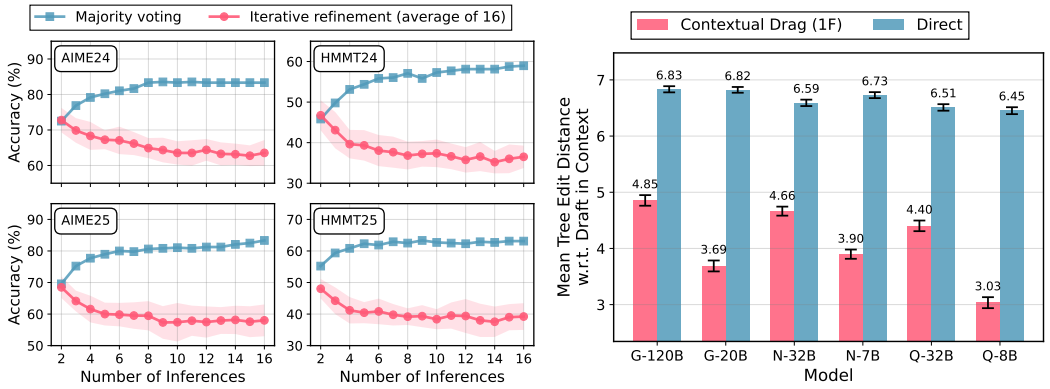
Compared to majority voting, where sampling multiple independent solutions steadily improves accuracy, iterative refinement results in a gradual decline. Therefore, contextual drag exposes such iterative pipelines to a critical vulnerability: without mechanisms to detect, filter, or reset from low-quality drafts in context, the model can become increasingly biased toward the erroneous context.

2.3 QUANTIFYING CONTEXTUAL DRAG: GAME OF 24

Given the in-context draft, the model clearly appears to be trying to craft a new solution. What explains the contextual drag? One plausible explanation is that the underlying architecture (e.g., the attention mechanism) predisposes the model to reuse some reasoning patterns from the draft while falling well short of full copying. We study this possibility on *Game of 24* (also often seen as a restricted variant of the *Countdown* problem), where the similarity between answers is quantifiable.

[2]We verify correctness by comparing the parsed final answer with ground truth via symbolic verifier (Kydlíček, 2024). The filter ensures the task is nontrivial yet not impossible to solve for the models.

[3]The model first generates an initial response with no additional context. Then the output from the previous inference is used as the in-context draft. Note that the draft can be either correct or incorrect in this setting.



(a) Self-deterioration

(b) Structural similarity measured by TED

Figure 2: **(a) Self-deterioration**: For models with severe contextual drag (e.g., GPT-OSS-20B), iterative refinement collapses in accuracy (average over 16 refinement trajectories) across iterations. **(b) Contextual drag arises from copying reasoning patterns**: Mean tree edit distance (TED) shows that conditioned 1F solutions remain closer to in-context reasoning than DIRECT solutions. Lower TED indicates stronger structural similarity, not necessarily worse performance.

Given four integers from $1, \ldots, 13$, the task is to form an expression using each exactly once with $+, -, \times, \div$ and parentheses to obtain 24. Representing solutions as trees with integers as leaves and operations as internal nodes allows us to measure structural similarity, and therefore the extent to which the models' subsequent answers resemble the incorrect drafts, via tree edit distance (TED) (Zhang & Shasha, 1989). For each problem, we pair the draft with clean-slate responses under and conditioned responses under contextual drag and compute the distance between each pair (Figure 6).

Figure 2b shows that the conditioned responses under 1F remain significantly closer to the incorrect draft than the clean-slate responses under DIRECT across models[4]. This demonstrates that contextual drag operates at the level of internal reasoning structure: models do not merely imitate surface tokens, but subtly follow the erroneous computational pathway in the draft. Additional results under 2F suggests that diversity of responses in context does not substantially alleviate the bias in structure (Appendix Figure 7). The results complement earlier sections by showing that contextual drag is not only a performance phenomenon but a systematic structural distortion of reasoning.

Overall, contextual drag represents a systematic failure mode of current LLMs with implications for multi-step reasoning workflows. Beyond iterative refinement, incorrect hypotheses and failed branches in extended CoTs can similarly bias subsequent exploration (Feng et al., 2025), resembling a form of contextual drag in extended reasoning. We leave further investigation to future work.

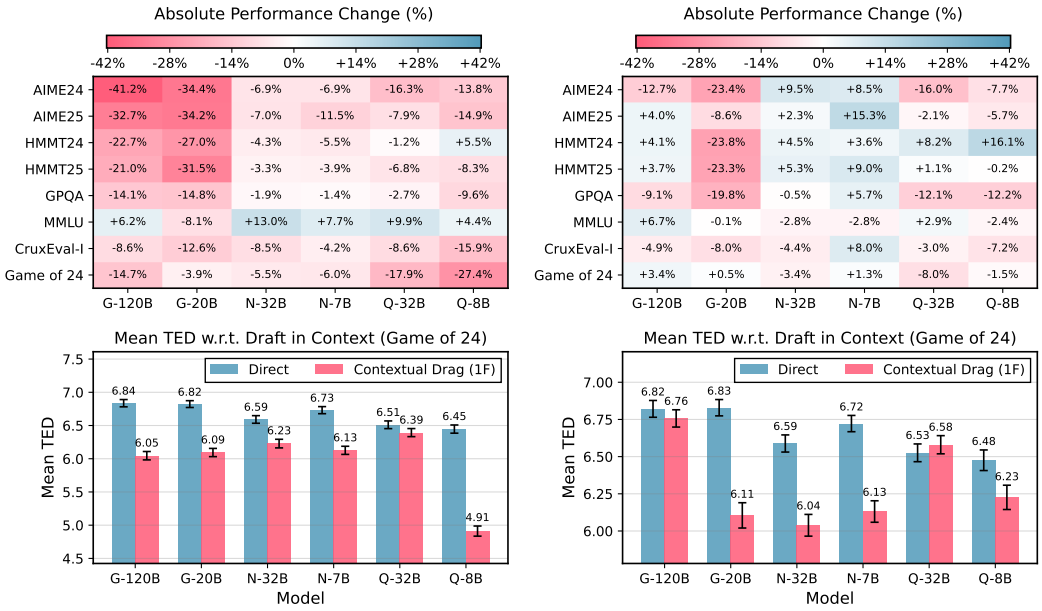
3 CONTEXTUAL DRAG UNDER ERROR SIGNALS

A natural hypothesis is that contextual drag is mainly a verification failure: once the model is given a reliable signal that the in-context draft is wrong, it should at least avoid similar errors when producing a new answer. In this section, we show that even with explicit, accurate error signals, either provided externally in the prompt (Section 3.1) or produced by the model (Section 3.2), conditioning on an incorrect draft can still bias subsequent reasoning.

3.1 EXTERNAL ERROR SIGNAL

To test whether contextual drag is mainly caused by the lack of error signals, we reuse the 1F evaluation setup in Section 2.1 with a modified prompt that explicitly highlights the incorrectness of the draft and warns against directly copying the answer (Appendix Section F). This provides an external, reliable error signal independent of the model's own verification capability.

[4]We highlight that TED is not a performance metric; it serves only as a qualitative indicator of structural similarity between conditioned generations and in-context drafts under contextual drag.



(a) Provided with external error signals

(b) Conditioned on self-detected error signals

Figure 3: **(a) External error signal does not recover subsequent reasoning from contextual drag**: Models still experience significant drops from DIRECT to 1F except on MMLU. See full results in Table 10. **(b) Correct self-verification gives rise to varying robustness to contextual drag**: Nemotron-7B/32B recover toward (and even surpass) DIRECT performance, while GPT-OSS-20B remains strongly degraded. See full results in Table 11.

As shown in Figure 3a, benchmark performance under 1F still drops substantially relative to DIRECT across tasks and models. Moreover, the generations under 1F on Game of 24 remain structurally closer to the incorrect draft than the clean-slate counterparts under DIRECT. Even with unambiguous error signals in the prompt, the draft continues to bias downstream reasoning. External error signals alone are insufficient to mitigate contextual drag.

3.2 SELF-DETECTED ERROR SIGNAL (POST-HOC)

We next examine the effect of successful self-verification, when the model itself explicitly recognizes that the draft is wrong, on contextual drag. We filter the 1F generations in Section 2.1 to only retain trajectories with an explicit correct verdict of the draft's incorrectness[5]. We note that this analysis is purely post-hoc and effectively assumes access to an oracle filter: it selectively keeps "good" trajectories and therefore biases toward stronger performance. We also note that the retained subset differs by model and task, the resulting accuracy and TED are not directly comparable to the full 1F results in Section 2.1.

As shown in Figure 3b, when conditioned on correct self-verification, structural contextual drag still generally persists on Game of 24. However, the impact on performance is no longer uniform across models. The improved behaviors of the Nemotron family are promising: it is possible for accurate self-verification to alleviate contextual drag and, in favorable cases, even surpass DIRECT performance. On the other hand, other models, especially GPT-OSS-20B, remain strongly degraded despite successful self-verification. The varying performance change indicates that verification may not be the only bottleneck for full recovery from contextual drag, presenting a great challenge to self-improvement pipelines that rely on external or self-generated feedback loops.

[5]We verify verdicts via parsable structured tags in the model output (Appendix Section H.3).

Table 2: **Context denoising mitigates but fails to eliminate contextual drag**: REV./REVISE and FILT./FILTER are shown. We include the VER ablation under 1F (denoted as 1F-V) for fair comparison (Table 9), as our implementation omits explicit verification in downstream inference.

	HMMT25*					GPQA-DIAMOND*					CRUXEVAL-I*				
	DIRECT	REV.	FILT.	1F	1F-V	DIRECT	REV.	FILT.	1F	1F-V	DIRECT	REV.	FILT.	1F	1F-V
GPT-OSS-120B	61.01	68.75	59.23	48.51	52.98	58.24	46.88	54.88	35.70	38.16	89.23	81.98	87.80	81.02	87.13
GPT-OSS-20B	58.63	62.20	45.83	20.54	42.26	46.97	33.33	43.70	16.86	26.66	61.34	61.24	49.73	42.39	45.11
Nemotron-32B	80.36	74.40	91.96	65.48	67.56	66.81	48.15	68.47	49.43	39.44	77.06	67.49	73.80	53.69	52.59
Nemotron-7B	63.10	63.10	62.20	54.17	51.19	51.61	35.27	55.73	35.27	27.46	43.55	48.34	39.13	35.27	35.34
Qwen3-32B	49.70	48.81	43.15	36.90	40.77	57.62	27.04	58.38	34.19	22.68	75.83	65.36	62.90	54.89	53.96
Qwen3-8B	42.26	38.69	42.86	34.23	36.61	47.77	25.24	45.27	27.13	18.75	67.49	49.00	51.80	40.09	38.33

4 MITIGATING CONTEXTUAL DRAG

Motivated by our finding that contextual drag persists even after error recognition (Section 3), we investigate methods that partially mitigate the impact of contextual drag from two perspectives in this section. First, models can be prompted to denoise the context content during test time (Section 4.1). Second, models can be trained via targeted supervised fine-tuning, for example, to ignore the in-context draft and switch to clean-slate reasoning upon error detection (Section 4.2).

4.1 CONTEXT DENOISING

We investigate two test-time context denoising pipelines: the model is asked to either (1) revise individual drafts to improve quality (REVISE) or; (2) filter the drafts to only keep the correct and useful intermediate steps given a high-level strategy produced by the model itself (FILTER). We include the complete details of the implementation in Section G.1.

4.1.1 RESULTS

Both methods mitigate, but do not eliminate, contextual drag, with FILTER being more robust. Across tasks and models, FILTER more consistently recovers the performance gap, while gains from REVISE are mixed. We attribute this to contextual drag persisting during the revision process of REVISE, whereas FILTER uses a clean-slate-generated high-level strategy to guide draft denoising. However, since both methods rely on the model's own, unreliable judgment of in-context drafts, neither fully recovers from the degradation in all settings. We include the full results in Section G.2.

Context denoising incurs higher inference costs at scale. Since both pipelines perform multiple inference passes per draft, test-time cost scales with the number of denoising steps and drafts. In practice, batching drafts can partially amortize this cost, but often at the expense of denoising quality.

4.2 TRAINING MODELS TO SWITCH TO CLEAN-SLATE REASONING ON ERROR DETECTION

We study a simple supervised mitigation that trains an explicit *fallback behavior*: after step-by-step verification, the model either reuses the draft that is deemed correct, or switches to a clean-slate solution trajectory when it detects an error in the draft's reasoning.

We fine-tune GPT-OSS-20B, which exhibits the largest drop even under error signals, with standard supervised learning. We sample 40k (problem, correct-draft) and 40k (problem, incorrect-draft) pairs from a moderately challenging subset of Big-Math-RL-Verified (Albalak et al., 2025). Drafts are drawn from a pool of open-weight reasoning models and verified with ground-truth answers.

For each (problem, incorrect-draft) pair, we concatenate (1) a verification reasoning trace generated by GPT-OSS-20B that correctly verifies the draft as "Incorrect" and (2) a clean-slate trace generated by the model for the same problem (Figure 9 1F). This provides an explicit fallback trace that does not reference the draft. For each pair of problem and correct draft, we concatenate (1) a verification trajectory that successfully verifies the draft as "Correct" and (2) a fixed template that guides the model to reuse the draft's answer (Figure 9 1T). This teaches a conditional reuse of helpful context rather than a uniform behavior of ignoring the draft. Complete details on data, prompts, and training are provided in Section H, including a comparison with GRPO in Section H.7.

Table 3: **Targeted SFT generally improves robustness under erroneous drafts**: Accuracy on competitive math benchmarks under 1F, and on the post-hoc subset where the model self-verifies the draft as "Incorrect" (Section 3.2). "Finetuned" denotes GPT-OSS-20B after our conditional fallback training; deltas in parentheses are absolute percentage-point changes vs. baseline.

SETTING	1F				1F (Conditioned on Self-Detected Error Signal)			
	AIME24*	AIME25*	HMMT24*	HMMT25*	AIME24*	AIME25*	HMMT24*	HMMT25*
Baseline	17.5	18.8	13.3	21.2	28.5	43.3	16.4	33.9
Finetuned	40.6 (+23.1)	20.7 (+1.92)	19.2 (+5.89)	27.8 (+6.56)	52.9 (+24.4)	38.7 (-4.65)	30.5 (+14.0)	45.3 (+11.4)

4.2.1 RESULTS

Targeted SFT consistently mitigates contextual drag under erroneous context, but does not fully restore clean-slate performance. As shown in Table 3, the lightweight synthesis rule via targeted SFT can measurably reduce the degradation under 1F. However, although the gap narrows, 1F accuracy remains below DIRECT, suggesting that residual contextual drag persists even after training the reset mechanism.

Targeted SFT improves the model's ability to act on self-detected error signals. We repeat the post-hoc conditioned analysis from Section 3.2, restricting the analysis to cases where the model correctly identifies the draft as incorrect. The fine-tuned model generally improves relative to its pre-tuning counterpart across benchmarks (except on AIME25). This provides a complementary view of training effectiveness: beyond improving unconditional 1F accuracy, targeted SFT strengthens the model's capacity to translate error awareness into improved downstream reasoning.

Improved robustness trades off with the use of correct context. Despite gains under erroneous context, targeted SFT incurs performance drops conditioning on correct drafts (Section H.6), revealing a robustness-utilization tradeoff: training the model to reset on error detection can also weaken its ability to exploit helpful context. These results position targeted SFT as a proof-of-concept mitigation, yet not a drop-in solution without sacrificing performance under correct context.

5 RELATED WORK

Context sensitivity and noisy CoTs Prior studies have documented that LLMs are sensitive to positional biases (Liu et al., 2024), framing (Lorè & Heydari, 2024), and verification (Asai et al., 2024) of contextual information. Meanwhile, providing noisy rationales in context can significantly degrade model performance, even when models are instructed to critique the flawed reasoning (Zhou et al., 2024; Li et al., 2025a; Lee et al., 2026). These findings suggest that LLMs are subject to strong influence from incorrect information in the context.

Self-improvement and multi-agent systems Sequential refinement (Madaan et al., 2023) and multi-agent aggregation (Madaan et al., 2025; Venkatraman et al., 2025; Wang et al., 2024a; Li et al., 2025c) pipelines implicitly assume that exposing the model to prior solutions provides informative guidance that facilitates the search for correct answers.

We are the first to show that the influence of erroneous context persists even after explicit error signals, and manifests as structural distortions in reasoning rather than solely as performance degradation. Models under severe contextual drag exhibit self-deterioration, posing a significant challenge to reliable self-improvement and multi-agent systems.

6 CONCLUSION

We identify contextual drag as a failure mode in reasoning models in which erroneous context biases subsequent reasoning toward structurally similar errors. This effect persists even with explicit error signals, indicating that it extends beyond verification and challenges iterative refinement pipelines. While targeted fine-tuning and context denoising partially mitigate this effect, neither restores clean-slate performance. See Appendix B for extended discussions. These findings highlight the need for principled mechanisms to reset or discount unreliable context in multi-step reasoning and self-improvement workflows.

REFERENCES

Wasi Uddin Ahmad, Somshubra Majumdar, Aleksander Ficek, Sean Narenthiran, Mehrzad Samadi, Jocelyn Huang, Siddhartha Jain, Vahid Noroozi, and Boris Ginsburg. Opencodereasoning-ii: A simple test time scaling approach via self-critique. *arXiv preprint arXiv:2507.09075*, 2025a.

Wasi Uddin Ahmad, Sean Narenthiran, Somshubra Majumdar, Aleksander Ficek, Siddhartha Jain, Jocelyn Huang, Vahid Noroozi, and Boris Ginsburg. Opencodereasoning: Advancing data distillation for competitive coding. *arXiv preprint arXiv:2504.01943*, 2025b.

Alon Albalak, Duy Phung, Nathan Lile, Rafael Rafailov, Kanishk Gandhi, Louis Castricato, Anikait Singh, Chase Blagden, Violet Xiang, Dakota Mahan, and Nick Haber. Big-math: A large-scale, high-quality math dataset for reinforcement learning in language models, 2025. URL https://arxiv.org/abs/2502.17387.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. 2024.

Mislav Balunović, Jasper Dekoninck, Ivo Petrov, Nikola Jovanović, and Martin Vechev. Matharena: Evaluating llms on uncontaminated math competitions, February 2025. URL https://matharena.ai/.

Shan Chen, Mingye Gao, Kuleen Sasse, Thomas Hartvigsen, Brian Anthony, Lizhou Fan, Hugo Aerts, Jack Gallifant, and Danielle S. Bitterman. When helpfulness backfires: Llms and the risk of false medical information due to sycophantic behavior. *npj Digital Medicine*, 8(1):605, Oct 2025. ISSN 2398-6352. doi: 10.1038/s41746-025-02008-z. URL https://doi.org/10.1038/s41746-025-02008-z.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Yunzhen Feng, Julia Kempe, Cheng Zhang, Parag Jain, and Anthony Hartshorn. What characterizes effective reasoning? revisiting length, review, and structure of cot. *arXiv preprint arXiv:2509.19284*, 2025.

Adrian Furnham and Hua Chu Boo. A literature review of the anchoring effect. *The Journal of Socio-Economics*, 40(1):35–42, 2011. ISSN 1053-5357. doi: https://doi.org/10.1016/j.socec.2010.10.008. URL https://www.sciencedirect.com/science/article/pii/S1053535710001411.

Aryo Pradipta Gema, Joshua Ong Jun Leang, Giwon Hong, Alessio Devoto, Alberto Carlo Maria Mancino, Rohit Saxena, Xuanli He, Yu Zhao, Xiaotang Du, Mohammad Reza Ghasemi Madani, Claire Barale, Robert McHardy, Joshua Harris, Jean Kaddour, Emile van Krieken, and Pasquale Minervini. Are we done with mmlu?, 2024.

Google. Gemini 2.5 pro model card. 2025a. URL https://storage.googleapis.com/deepmind-media/Model-Cards/Gemini-2-5-Pro-Model-Card.pdf.

Google. Gemini 3 pro model card. 2025b. URL https://storage.googleapis.com/deepmind-media/Model-Cards/Gemini-3-Pro-Model-Card.pdf.

Sachin Goyal, Christina Baek, J Zico Kolter, and Aditi Raghunathan. Context-parametric inversion: Why instruction finetuning may not actually improve context reliance. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=SPS6HzVzyt.

Alex Gu, Baptiste Rozière, Hugh Leather, Armando Solar-Lezama, Gabriel Synnaeve, and Sida I. Wang. Cruxeval: A benchmark for code reasoning, understanding and execution. *arXiv preprint arXiv:2401.03065*, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, Sep 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-09422-z. URL https://doi.org/10.1038/s41586-025-09422-z.

Zhonghao He, Tianyi Qiu, Hirokazu Shirado, and Maarten Sap. Martingale score: An unsupervised metric for bayesian rationality in LLM reasoning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=BfO6od6JD6.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=IkmD3fKBPQ.

Yiming Huang, Biquan Bie, Zuqiu Na, Weilin Ruan, Songxin Lei, Yutao Yue, and Xinlei He. An empirical study of the anchoring effect in llms: Existence, mechanism, and potential mitigations. *arXiv preprint arXiv:2505.15392*, 2025.

Zhuoran Jin, Pengfei Cao, Hongbang Yuan, Yubo Chen, Jiexin Xu, Huaijun Li, Xiaojian Jiang, Kang Liu, and Jun Zhao. Cutting off the head ends the conflict: A mechanism for interpreting and mitigating knowledge conflicts in language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 1193–1215, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.70. URL https://aclanthology.org/2024.findings-acl.70/.

Daniel Kahneman. *Thinking, Fast and Slow*. Farrar, Straus and Giroux, New York, 2011.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Hynek Kydlíček. math-verify: A library for verifying mathematical answers. https://github.com/huggingface/Math-Verify, 2024.

Seongyun Lee, Yongrae Jo, Minju Seo, Moontae Lee, and Minjoon Seo. Lost in the noise: How reasoning models fail with contextual distractors. *arXiv preprint arXiv:2601.07226*, 2026.

Chunyang Li, Weiqi Wang, Tianshi Zheng, and Yangqiu Song. Patterns over principles: The fragility of inductive reasoning in LLMs under noisy observations. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 19608–19626, Vienna, Austria, July 2025a. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.1006. URL https://aclanthology.org/2025.findings-acl.1006/.

Gaotang Li, Yuzhong Chen, and Hanghang Tong. Taming knowledge conflicts in language models. In *Forty-second International Conference on Machine Learning*, 2025b. URL https://openreview.net/forum?id=0cEZyhHEks.

Wenzhe Li, Yong Lin, Mengzhou Xia, and Chi Jin. Rethinking mixture-of-agents: Is mixing different large language models beneficial? *arXiv preprint arXiv:2502.00674*, 2025c.

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*, 12:157–173, 2024.

Nunzio Lorè and Babak Heydari. Strategic behavior of large language models and the role of game structure versus contextual framing. *Scientific Reports*, 14(1):18490, Aug 2024. ISSN 2045-2322. doi: 10.1038/s41598-024-69032-z. URL https://doi.org/10.1038/s41598-024-69032-z.

Jiaxu Lou and Yifan Sun. Anchoring bias in large language models: An experimental study. *Journal of Computational Social Science*, 9(1):11, 2026.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36:46534–46594, 2023.

Lovish Madaan, Aniket Didolkar, Suchin Gururangan, John Quan, Ruan Silva, Ruslan Salakhutdinov, Manzil Zaheer, Sanjeev Arora, and Anirudh Goyal. Rethinking thinking tokens: Llms as improvement operators. *arXiv preprint arXiv:2510.01123*, 2025.

Ivan Moshkov, Darragh Hanley, Ivan Sorokin, Shubham Toshniwal, Christof Henkel, Benedikt Schifferer, Wei Du, and Igor Gitman. Aimo-2 winning solution: Building state-of-the-art mathematical reasoning models with openmathreasoning dataset. *arXiv preprint arXiv:2504.16891*, 2025.

Jeremy K. Nguyen. Human bias in ai models? anchoring effects and mitigation strategies in large language models. *Journal of Behavioral and Experimental Finance*, 43:100971, 2024. ISSN 2214-6350. doi: https://doi.org/10.1016/j.jbef.2024.100971. URL https://www.sciencedirect.com/science/article/pii/S2214635024000868.

OpenAI. Learning to reason with llms. 2024. URL https://openai.com/index/learning-to-reason-with-llms/.

OpenAI. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025a.

OpenAI. Gpt-5 system card. 2025b. URL https://cdn.openai.com/gpt-5-system-card.pdf.

Daniel E O'Leary. An anchoring effect in large language models. *IEEE Intelligent Systems*, 40(2): 23–26, 2025a.

Daniel E O'Leary. Confirmation and Specificity Biases in Large Language Models: An Explorative Study . *IEEE Intelligent Systems*, 40(01):63–68, January 2025b. ISSN 1941-1294. doi: 10.1109/MIS.2024.3513992. URL https://doi.ieeecomputersociety.org/10.1109/MIS.2024.3513992.

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.

Mrinank Sharma, Meg Tong, Tomasz Korbak, David Duvenaud, Amanda Askell, Samuel R. Bowman, Esin DURMUS, Zac Hatfield-Dodds, Scott R Johnston, Shauna M Kravec, Timothy Maxwell, Sam McCandlish, Kamal Ndousse, Oliver Rausch, Nicholas Schiefer, Da Yan, Miranda Zhang, and Ethan Perez. Towards understanding sycophancy in language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=tvhaxkMKAn.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel Bowman. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. *Advances in Neural Information Processing Systems*, 36:74952–74965, 2023.

Amos Tversky and Daniel Kahneman. Judgment under uncertainty: Heuristics and biases: Biases in judgments reveal some heuristics of thinking under uncertainty. *science*, 185(4157):1124–1131, 1974.

Siddarth Venkatraman, Vineet Jain, Sarthak Mittal, Vedant Shah, Johan Obando-Ceron, Yoshua Bengio, Brian R Bartoldson, Bhavya Kailkhura, Guillaume Lajoie, Glen Berseth, et al. Recursive self-aggregation unlocks deep thinking in large language models. *arXiv preprint arXiv:2509.26626*, 2025.

Junlin Wang, Jue Wang, Ben Athiwaratkun, Ce Zhang, and James Zou. Mixture-of-agents enhances large language model capabilities. *arXiv preprint arXiv:2406.04692*, 2024a.

Yike Wang, Shangbin Feng, Heng Wang, Weijia Shi, Vidhisha Balachandran, Tianxing He, and Yulia Tsvetkov. Resolving knowledge conflicts in large language models. In *First Conference on Language Modeling*, 2024b. URL https://openreview.net/forum?id=ptvV5HGTNN.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit bayesian inference. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=RdJVFCHjUMI.

Wenda Xu, Guanglei Zhu, Xuandong Zhao, Liangming Pan, Lei Li, and William Wang. Pride and prejudice: Llm amplifies self-bias in self-refinement. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15474–15492, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Kaizhong Zhang and Dennis Shasha. Simple fast algorithms for the editing distance between trees and related problems. *SIAM Journal on Computing*, 18(6):1245–1262, 1989. doi: 10.1137/0218082. URL https://doi.org/10.1137/0218082.

Jun Zhao, Yongzhuo Yang, Xiang Hu, Jingqi Tong, Yi Lu, Wei Wu, Tao Gui, Qi Zhang, and Xuanjing Huang. Understanding parametric and contextual knowledge reconciliation within large language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=76cFMRgEzQ.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand, 2024. Association for Computational Linguistics. URL http://arxiv.org/abs/2403.13372.

Zhanke Zhou, Rong Tao, Jianing Zhu, Yiwen Luo, Zengmao Wang, and Bo Han. Can language models perform robust reasoning in chain-of-thought prompting with noisy rationales? *Advances in Neural Information Processing Systems*, 37:123846–123910, 2024.

Wang Bill Zhu, Tianqi Chen, Ching Ying Lin, Jade Law, Mazen Jizzini, Jorge J Nieva, Ruishan Liu, and Robin Jia. Cancer-myth: Evaluating ai chatbot on patient questions with false presuppositions. *arXiv preprint arXiv:2504.11373*, 2025.

A Related Work

Anchoring bias LLMs exhibit anchoring behavior, which mirrors the anchoring effect in human judgment and decision-making (Tversky & Kahneman, 1974; Furnham & Boo, 2011). LLM outputs can be systematically biased by numerical hints or framing information embedded in prompts (Nguyen, 2024; Huang et al., 2025; O'Leary, 2025a;b; Lou & Sun, 2026). These works establish anchoring as a reproducible phenomenon in modern LLMs.

Context sensitivity and noisy CoTs Prior studies have documented that LLMs are sensitive to positional biases (Liu et al., 2024), framing (Lorè & Heydari, 2024), and verification (Asai et al., 2024) of contextual information. Meanwhile, providing noisy rationales in context can significantly degrade model performance, even when models are instructed to critique the flawed reasoning (Zhou et al., 2024; Li et al., 2025a; Lee et al., 2026). These findings suggest that LLMs are subject to strong influence from incorrect information in the context.

Self-improvement and multi-agent systems Sequential refinement (Madaan et al., 2023) and multi-agent aggregation (Madaan et al., 2025; Venkatraman et al., 2025; Wang et al., 2024a; Li et al., 2025c) pipelines implicitly assume that exposing the model to prior solutions provides informative guidance that facilitates the search for correct answers.

Conflicts between context and parametric knowledge Recent research examines how LLMs resolve conflicts between internal parametric knowledge and non-parametric information in the context (Zhao et al., 2025; Goyal et al., 2025), exploring interventions through head pruning (Jin et al., 2024), fine-tuning (Wang et al., 2024b), and attention interventions (Li et al., 2025b). Context-parametric knowledge conflicts are primarily defined at the level of factual or semantic correctness.

Contextual drag in our study can be viewed as a form of anchoring in reasoning, but we distinguish our work from prior work in several key ways. We are the first to show that the influence of erroneous context persists even after explicit error signals, and manifests as structural distortions in reasoning rather than solely as performance degradation. Models under severe contextual drag exhibit self-deterioration, posing a significant challenge to reliable self-improvement and multi-agent systems. Our findings also complement work on belief persistence in knowledge conflicts (He et al., 2025).

B Limitations and Future Work

Contextual drag reflects a failure mode of in-context learning where models over-exploit coherent but incorrect context. This reliance, which emerges during pretraining on structured, high-quality data (Xie et al., 2022), explains why correct drafts produce large gains while incorrect ones cause degradation. Our targeted training shows a tradeoff between coherence-driven context use and robustness to errors: reducing contextual drag often diminishes benefits from correct context. Understanding how this tradeoff emerges from training choices remains an open question.

We do not perform an extensive mechanistic study of contextual drag. As suggested by our structural analysis in Section 2.3, the effect primarily manifests in shifts in the reasoning structure rather than localized token-level behaviors. Compared to prior work that focus on simpler tasks with short outputs (e.g., single-word answers) where attention patterns can be more easily traced (Huang et al., 2025), the long reasoning trajectories in our setting make it substantially harder to isolate specific mechanisms in representation space.

We also note model-dependent recovery after correct self-verification between Nemotron models and GPT-OSS-20B (Section 3.2). Fine-grained comparisons may uncover mechanisms, such as reasoning-trace distillation, that enhance robustness to contextual drag. Finally, the persistence of the effect suggests a limitation of attention-based architectures, which lack mechanisms to fully reset the reasoning state despite verification. Addressing this issue may require architectural or training-level advances that enable more selective use of context.

C Additional Details of Contextual Drag Evaluation

C.1 Prompt Templates

We provide the prompt templates used to evaluate contextual drag in Table 1 below:

DIRECT: given the problem, the model is asked to solve the problem with chain-of-thoughts;

DIRECT

```
{problem} Let's think step by step and please wrap your final answer in
\boxed{}.
```

1F/2F: given the problem and one or two incorrect model-generated draft solutions, the model is asked to check the correctness of the draft and generate a new solution. We verify correctness by parsing the final answer and compare the model's answer with ground truth via symbolic verifier Kydlíček (2024). We remove all the thinking traces wrapped in `<think></think>` from the drafts to keep the context length manageable.

1F

```
In this task, you will be provided with a problem and one draft
solution to that problem.  For the draft, walk through the solution
step-by-step and check for any fatal mistakes in the reasoning (for
example:  incorrect use of theorems or facts, incorrect calculations,
etc.).  Please note that once you spot a fatal error in a draft's
reasoning, you **MUST** stop checking the remaining steps of that draft
and move on.

For the draft, output an overall verdict on the correctness of the
draft using the following tag format:

<overall_verdict>[Correct / Incorrect]</overall_verdict>

While evaluating individual steps, be strict about algebra, logic,
and correct theorem application.  If the draft is incorrect, write a
complete, correct solution (you may reuse sound ideas from the draft).
If the draft is correct, you may adopt and polish it as the final
solution.

Conclude by presenting the final answer only inside \boxed{} (i.e., the
final answer should appear only inside a LaTeX `\boxed{}`).

The problem is as follows:
-- beginning of problem --
{problem}
-- end of problem --

Here is the draft solution you need to consider:

### Solution:
-- beginning of the draft --
{draft1}
-- end of the draft --

Please:
1.  Carefully evaluate the draft solution step-by-step.  Once you spot
a fatal error in a draft's reasoning, you **MUST** stop checking the
remaining steps of that draft and move on.
2.  Output the correctness verdicts for the draft.
3.  Provide a correct solution to the problem with complete reasoning
steps that lead to the answer.

Remember:  conclude with the final answer only in \boxed{}.
```

2F

```
In this task, you will be provided with a problem and two draft
solutions to that problem.  For each draft, walk through the solution
step-by-step and check for any fatal mistakes in the reasoning (for
example:  incorrect use of theorems or facts, incorrect calculations,
etc.).  Please note that once you spot a fatal error in a draft's
reasoning, you **MUST** stop checking the remaining steps of that draft
and move on to the next draft.

For each draft (Solution 1 and Solution 2), output an overall verdict
on the correctness of the draft using the following tag format:

<overall_verdict_1>[Correct / Incorrect]</overall_verdict_1>
<overall_verdict_2>[Correct / Incorrect]</overall_verdict_2>

While evaluating individual steps, be strict about algebra, logic,
and correct theorem application.  If both drafts are incorrect, write
a complete, correct solution (you may reuse sound ideas from the
drafts).  If a draft is correct, you may adopt and polish it as the
final solution.

Conclude by presenting the final answer only inside \boxed{} (i.e., the
final answer should appear only inside a LaTeX `\boxed{}`).

The problem is as follows:
-- beginning of problem --
{problem}
-- end of problem --

Here are the two draft solutions you need to consider:

### Solution 1:
-- beginning of the first draft --
{draft1}
-- end of the first draft --

### Solution 2:
-- beginning of the second draft --
{draft2}
-- end of the second draft --

Please:
1.  Carefully evaluate the two draft solutions step-by-step.  Once you
spot a fatal error in a draft's reasoning, you **MUST** stop checking
the remaining steps of that draft and move on to the next draft.
2.  Output the correctness verdicts for the drafts, respectively.
3.  Provide a correct solution to the problem with complete reasoning
steps that lead to the answer.

Remember:  conclude with the final answer only in \boxed{}.
```

Figure 4 illustrates the evaluation setup for contextual drag in Table 1.

C.2 Inference Configurations

We followed the official recommended generation configurations for model inference, as listed in Table 4. For efficient inference, we use vLLM project version 0.10.2 Kwon et al. (2023). We use a total context size of $32{,}768$ for all models except for the Nemotron family, which has a recommended max output token limit of $65{,}536$. As later shown in Table 5, this does not give extra advantages to the Nemotron models as all open-weight models are able to finish most of generations within the specified max token limits with comparable completion rates.

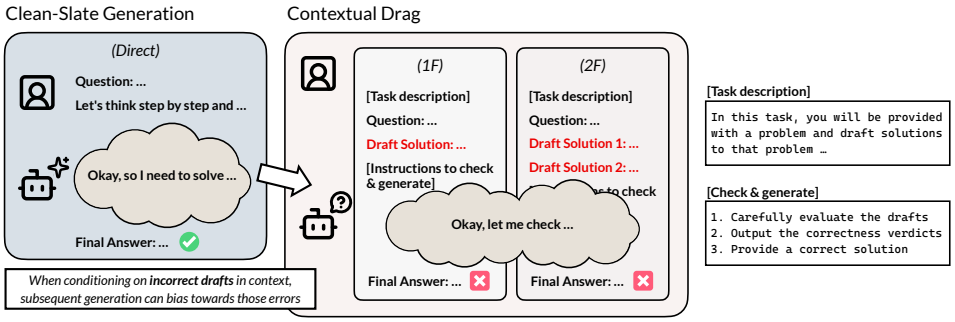


Figure 4: **Contextual drag evaluation setup**: We measure (1) DIRECT performance of the model's clean-slate generation, and (2) 1F/2F performance of the model's generation conditioned on 1 or 2 incorrect draft solutions. DIRECT evaluation uses the standard chain-of-thought prompting. For 1F/2F evaluations, the models is instructed to perform a step-by-step verification then produce a correctness verdict for each in-context draft, then produce a new solution.

Table 4: Inference configurations for all evaluated models. We report the maximum context length and the sampling parameters used during generation. For models where `top_k` sampling is not being used, we denote it as "–".

Model	#tokens	Temp	Top-p / Top-k	Huggingface Path
Qwen3-8B (Thinking)	32,768	0.6	0.95 / 20	`Qwen/Qwen3-8B`
Qwen3-32B (Thinking)	32,768	0.6	0.95 / 20	`Qwen/Qwen3-32B`
Nemotron-7B	65,536	0.6	0.95 / –	`nvidia/OpenReasoning-Nemotron-7B`
Nemotron-32B	65,536	0.6	0.95 / –	`nvidia/OpenReasoning-Nemotron-32B`
GPT-OSS-20B	32,768	1.0	1.00 / 40	`openai/gpt-oss-20b`
GPT-OSS-120B	32,768	1.0	1.00 / 40	`openai/gpt-oss-120b`
LlamaR1-8B	32,768	0.6	0.95 / –	`deepseek-ai/DeepSeek-R1-Distill-Llama-8B`
QwenR1-7B	32,768	0.6	0.95 / –	`deepseek-ai/DeepSeek-R1-Distill-Qwen-7B`

C.3 INITIAL EVALUATION: SELECTION OF ANCHOR MODELS, IN-CONTEXT DRAFTS, AND EVALUATION PROBLEMS

Our criteria for constructing the evaluation set for contextual drag is to avoid confounding performance drops caused by trivial, low-quality in-context drafts observed by Li et al. (2025c). To this end, we choose a set of three *anchor models*, which have the top-3 clean-slate performance among all open-weight models for each task.

Note that under this setup, the in-context draft used in the contextual drag evaluation may either come from the evaluated model itself or from other models, depending on whether the evaluated model is chosen as one of the anchor models. Therefore, the evaluation reflects contextual drag in both self-refinement and multi-agent aggregation settings.

Finally, we select the evaluation problems based on their *reasonable hardness* to represent the most realistic use case in test-time scaling, such as iterative refinement. Therefore, our criterion for this evaluation subset is the set of questions where the anchor models exhibit both successes and failures: each selected question has at least two correct and at least two incorrect clean-slate responses from the anchor models. We note that these subsets are also of the most interest for test-time scaling as they correspond to the gap between pass@1 and pass@k (for larger k).

Table 5 shows the initial evaluation results of the clean-slate performance, which gives the following lists of anchor models:

Table 5: **Initial evaluation results of open-weight models on all evaluation benchmarks**: We first run clean-slate generation for the set of open-weight models on the original data of all benchmarks to select the top-3 performant models (underlined), whose responses are used as the in-context drafts for later evaluation of contextual drag.

Model	AIME24	AIME25	HMMT24	HMMT25
GPT-OSS-120B	79.79 (±0.45)	78.54 (±0.41)	52.92 (±0.48)	56.04 (±0.41)
GPT-OSS-20B	72.71 (±0.46)	68.33 (±0.60)	46.88 (±0.41)	54.37 (±0.46)
Nemotron-32B	87.92 (±0.35)	83.75 (±0.35)	59.79 (±0.54)	69.79 (±0.34)
Nemotron-7B	81.88 (±0.30)	79.17 (±0.36)	52.71 (±0.49)	57.50 (±0.55)
Qwen3-32B	80.00 (±0.33)	71.67 (±0.45)	47.29 (±0.51)	48.12 (±0.48)
Qwen3-8B	75.00 (±0.28)	67.08 (±0.39)	38.12 (±0.28)	42.92 (±0.39)
LlamaR1-8B	43.33 (±0.71)	32.29 (±0.42)	23.54 (±0.31)	20.21 (±0.39)
QwenR1-7B	54.17 (±0.71)	36.25 (±0.48)	28.96 (±0.46)	26.67 (±0.47)
Model	GPQA-DIAMOND	MMLU-REDUX	CRUXEVAL-I	GAME OF 24
GPT-OSS-120B	69.35 (±0.07)	90.90 (±0.00)	89.51 (±0.07)	85.95 (±0.01)
GPT-OSS-20B	58.59 (±0.08)	87.05 (±0.00)	71.97 (±0.21)	86.28 (±0.02)
Nemotron-32B	75.19 (±0.04)	90.87 (±0.00)	93.27 (±0.01)	89.96 (±0.01)
Nemotron-7B	62.63 (±0.07)	84.24 (±0.00)	55.13 (±0.11)	87.02 (±0.01)
Qwen3-32B	69.16 (±0.06)	91.64 (±0.00)	94.11 (±0.01)	89.15 (±0.01)
Qwen3-8B	60.64 (±0.07)	88.66 (±0.00)	90.91 (±0.01)	88.11 (±0.01)
LlamaR1-8B	42.87 (±0.08)	66.72 (±0.01)	56.16 (±0.02)	52.41 (±0.02)
QwenR1-7B	34.69 (±0.04)	61.19 (±0.01)	67.57 (±0.02)	75.19 (±0.01)

Table 6: List of anchor models whose clean-slate responses are used as in-context drafts in the evaluation of contextual drag in Table 1.

Task	Anchor Models
AIME24	Nemotron-7B/32B, Qwen3-32B
AIME25	GPT-OSS-120B, Nemotron-7B/32B
HMMT24	GPT-OSS-120B, Nemotron-7B/32B
HMMT25	GPT-OSS-120B, Nemotron-7B/32B
GPQA-DIAMOND	GPT-OSS-120B, Nemotron-32B, Qwen3-32B
MMLU-REDUX	GPT-OSS-120B, Nemotron-32B, Qwen3-32B
CRUXEVAL-I	Nemotron-32B, Qwen3-8B/32B
GAME OF 24	Nemotron-32B, Qwen3-8B/32B

C.4 FULL RESULTS OF CONTEXTUAL DRAG EVALUATION

Table 8 and Table 7 present the full pass@1 and pass@5 evaluation results for Table 1. We conduct 16 independent inferences for each problem instance. We report the relative performance changes with respect to DIRECT performance and provide the confidence intervals for the mean performance.

We note that there are no substantial differences between the pass@1 and pass@5 results, demonstrating that contextual drag leads to strong, consistent performance degradation that is hard to recover from increasing number of independent samplings.

Table 7: Full average pass@5 results (with 95% confidence interval in subscripts) for contextual drag evaluation in Table 1.

	AIME24*			AIME25*		
	DIRECT	1F	2F	DIRECT	1F	2F
GPT-OSS-120B	90.62 ±1.02	68.75 ±0.75 (-21.88)	73.12 ±0.90 (-17.50)	92.31 ±0.74	83.65 ±0.43 (-8.65)	74.52 ±0.88 (-17.79)
GPT-OSS-20B	83.12 ±1.31	40.00 ±0.55 (-43.12)	46.25 ±1.44 (-36.88)	78.85 ±0.69	58.65 ±1.33 (-20.19)	62.98 ±1.44 (-15.87)
Nemotron-32B	100.00 ±0.00	94.38 ±0.60 (-5.63)	92.50 ±1.40 (-7.50)	91.83 ±0.25	88.94 ±0.31 (-2.88)	77.40 ±0.78 (-14.42)
Nemotron-7B	89.38 ±0.38	78.75 ±0.51 (-10.63)	88.12 ±1.13 (-1.25)	87.50 ±0.63	86.54 ±0.52 (-0.96)	63.46 ±0.87 (-24.04)
Qwen3-32B	98.75 ±0.51	67.50 ±0.67 (-31.25)	56.88 ±0.72 (-41.88)	79.33 ±0.48	63.46 ±0.83 (-15.87)	54.33 ±0.45 (-25.00)
Qwen3-8B	76.88 ±1.06	63.75 ±0.51 (-13.12)	54.37 ±1.64 (-22.50)	68.75 ±0.69	51.44 ±0.25 (-17.31)	34.62 ±1.17 (-34.13)
LlamaR1-8B	48.12 ±1.57	26.88 ±0.38 (-21.25)	13.75 ±0.75 (-34.38)	23.08 ±0.64	13.46 ±0.37 (-9.62)	14.42 ±1.38 (-8.65)
QwenR1-7B	59.38 ±0.38	38.75 ±0.64 (-20.62)	31.87 ±2.46 (-27.50)	28.85 ±0.69	18.27 ±0.43 (-10.58)	4.81 ±0.51 (-24.04)
	HMMT24*			HMMT25*		
	DIRECT	1F	2F	DIRECT	1F	2F
GPT-OSS-120B	87.11 ±0.57	68.36 ±0.40 (-18.75)	75.00 ±0.61 (-12.11)	87.20 ±0.50	72.62 ±0.31 (-14.58)	66.67 ±0.48 (-20.54)
GPT-OSS-20B	80.86 ±0.50	40.23 ±0.49 (-40.62)	47.66 ±1.24 (-33.20)	86.01 ±0.85	52.98 ±0.35 (-33.04)	54.76 ±0.48 (-31.25)
Nemotron-32B	89.84 ±0.53	82.81 ±0.33 (-7.03)	85.16 ±0.53 (-4.69)	96.43 ±0.22	86.31 ±0.17 (-10.12)	85.12 ±0.31 (-11.31)
Nemotron-7B	87.11 ±0.63	64.84 ±0.57 (-22.27)	71.48 ±0.71 (-15.62)	91.37 ±0.32	78.57 ±0.25 (-12.80)	78.87 ±0.40 (-12.50)
Qwen3-32B	69.92 ±0.56	63.67 ±0.50 (-6.25)	60.94 ±0.33 (-8.98)	72.92 ±0.47	55.95 ±0.13 (-16.96)	63.69 ±0.40 (-9.23)
Qwen3-8B	51.95 ±0.59	49.22 ±0.57 (-2.73)	51.56 ±0.43 (-0.39)	63.99 ±0.74	51.19 ±0.40 (-12.80)	55.95 ±0.38 (-8.04)
LlamaR1-8B	19.14 ±0.69	21.88 ±0.27 (+2.73)	28.91 ±0.46 (+9.77)	24.70 ±0.63	20.83 ±0.17 (-3.87)	28.27 ±0.55 (+3.57)
QwenR1-7B	27.73 ±0.94	26.56 ±0.51 (-1.17)	25.00 ±0.81 (-2.73)	48.81 ±0.64	29.46 ±0.22 (-19.35)	36.61 ±0.74 (-12.20)
	GPQA-DIAMOND*			MMLU-REDUX*		
	DIRECT	1F	2F	DIRECT	1F	2F
GPT-OSS-120B	84.23 ±0.08	60.27 ±0.05 (-23.96)	53.93 ±0.08 (-30.30)	87.52 ±0.01	62.34 ±0.01 (-25.19)	49.91 ±0.02 (-37.61)
GPT-OSS-20B	80.07 ±0.05	42.90 ±0.08 (-37.17)	32.10 ±0.11 (-47.96)	85.31 ±0.02	47.24 ±0.02 (-38.07)	30.00 ±0.02 (-55.32)
Nemotron-32B	90.34 ±0.04	69.93 ±0.07 (-20.41)	55.26 ±0.09 (-35.09)	87.83 ±0.01	50.74 ±0.01 (-37.09)	32.28 ±0.01 (-55.55)
Nemotron-7B	79.64 ±0.08	59.14 ±0.03 (-20.50)	46.97 ±0.10 (-32.67)	80.72 ±0.02	43.15 ±0.01 (-37.58)	26.93 ±0.01 (-53.79)
Qwen3-32B	79.97 ±0.07	54.31 ±0.04 (-25.66)	37.22 ±0.05 (-42.76)	86.91 ±0.01	61.70 ±0.01 (-25.21)	36.99 ±0.02 (-49.91)
Qwen3-8B	66.76 ±0.09	45.03 ±0.05 (-21.73)	30.40 ±0.05 (-36.36)	77.58 ±0.01	42.18 ±0.01 (-35.40)	29.82 ±0.01 (-47.76)
LlamaR1-8B	64.82 ±0.12	35.94 ±0.07 (-28.88)	27.89 ±0.06 (-36.93)	71.67 ±0.02	30.23 ±0.02 (-41.44)	18.50 ±0.01 (-53.17)
QwenR1-7B	41.71 ±0.08	24.62 ±0.05 (-17.09)	23.82 ±0.08 (-17.90)	64.59 ±0.03	29.54 ±0.02 (-35.05)	21.30 ±0.01 (-43.29)
	CRUXEVAL-I*			GAME OF 24*		
	DIRECT	1F	2F	DIRECT	1F	2F
GPT-OSS-120B	99.14 ±0.01	94.41 ±0.02 (-4.72)	93.28 ±0.03 (-5.85)	97.43 ±0.01	83.65 ±0.02 (-13.77)	84.52 ±0.02 (-12.90)
GPT-OSS-20B	92.15 ±0.03	75.00 ±0.02 (-17.15)	72.31 ±0.05 (-19.85)	98.76 ±0.01	72.78 ±0.03 (-25.98)	73.31 ±0.02 (-25.45)
Nemotron-32B	93.35 ±0.03	78.56 ±0.01 (-14.79)	74.83 ±0.04 (-18.52)	95.65 ±0.01	86.54 ±0.01 (-9.10)	83.43 ±0.01 (-12.21)
Nemotron-7B	80.52 ±0.06	64.49 ±0.06 (-16.02)	60.94 ±0.08 (-19.58)	94.71 ±0.01	80.84 ±0.01 (-13.87)	72.16 ±0.02 (-22.55)
Qwen3-32B	94.05 ±0.02	78.49 ±0.02 (-15.56)	74.27 ±0.04 (-19.78)	97.51 ±0.01	77.56 ±0.02 (-19.96)	51.71 ±0.03 (-45.80)
Qwen3-8B	85.17 ±0.03	61.10 ±0.03 (-24.07)	59.74 ±0.09 (-25.43)	95.33 ±0.01	59.60 ±0.01 (-35.73)	46.41 ±0.03 (-48.92)
LlamaR1-8B	56.85 ±0.06	45.15 ±0.03 (-11.70)	41.32 ±0.06 (-15.53)	69.55 ±0.03	15.97 ±0.01 (-53.58)	9.37 ±0.02 (-60.18)
QwenR1-7B	66.39 ±0.09	43.55 ±0.03 (-22.84)	34.97 ±0.03 (-31.42)	74.44 ±0.03	39.18 ±0.02 (-35.26)	23.43 ±0.02 (-51.01)

Table 8: Full average pass@1 results (with 95% confidence interval in subscripts) for contextual drag evaluation in Table 1.

	AIME24*			AIME25*		
	DIRECT	1F	2F	DIRECT	1F	2F
GPT-OSS-120B	66.25 ±1.72	43.75 ±0.51 (-22.50)	43.75 ±2.04 (-22.50)	66.83 ±1.47	55.29 ±1.04 (-11.54)	40.38 ±1.31 (-26.44)
GPT-OSS-20B	51.88 ±1.47	17.50 ±1.60 (-34.38)	21.25 ±2.32 (-30.63)	51.92 ±1.55	18.75 ±0.84 (-33.17)	20.67 ±1.51 (-31.25)
Nemotron-32B	83.75 ±1.81	67.50 ±1.28 (-16.25)	63.13 ±1.42 (-20.62)	78.37 ±1.18	61.54 ±0.91 (-16.83)	50.96 ±1.47 (-27.40)
Nemotron-7B	67.50 ±1.28	53.12 ±1.22 (-14.38)	58.75 ±1.81 (-8.75)	67.79 ±1.12	52.40 ±0.69 (-15.38)	37.02 ±1.29 (-30.77)
Qwen3-32B	65.00 ±2.05	41.25 ±0.84 (-23.75)	30.00 ±1.55 (-35.00)	54.81 ±1.47	31.25 ±0.66 (-23.56)	31.25 ±0.78 (-23.56)
Qwen3-8B	55.62 ±2.12	30.63 ±0.90 (-25.00)	20.62 ±1.69 (-35.00)	45.67 ±1.20	20.19 ±0.77 (-25.48)	11.06 ±1.04 (-34.62)
LlamaR1-8B	21.25 ±1.89	8.13 ±0.94 (-13.12)	4.38 ±0.77 (-16.88)	7.21 ±0.86	4.33 ±0.55 (-2.88)	2.88 ±0.63 (-4.33)
QwenR1-7B	33.13 ±2.37	11.88 ±1.09 (-21.25)	8.12 ±1.25 (-25.00)	9.62 ±0.78	3.85 ±0.37 (-5.77)	0.96 ±0.35 (-8.65)
	HMMT24*			HMMT25*		
	DIRECT	1F	2F	DIRECT	1F	2F
GPT-OSS-120B	48.83 ±1.22	39.84 ±0.60 (-8.98)	45.31 ±1.07 (-3.52)	61.01 ±0.70	48.51 ±0.55 (-12.50)	49.11 ±0.69 (-11.90)
GPT-OSS-20B	40.23 ±1.08	13.28 ±0.83 (-26.95)	17.97 ±1.21 (-22.27)	58.63 ±0.78	20.54 ±0.68 (-38.10)	24.70 ±1.02 (-33.93)
Nemotron-32B	61.72 ±1.40	51.95 ±0.54 (-9.77)	50.78 ±0.85 (-10.94)	80.36 ±0.62	65.48 ±0.55 (-14.88)	64.58 ±0.70 (-15.77)
Nemotron-7B	49.22 ±1.14	35.94 ±0.66 (-13.28)	40.62 ±0.77 (-8.59)	63.10 ±0.94	54.17 ±0.31 (-8.93)	52.38 ±0.74 (-10.71)
Qwen3-32B	39.06 ±1.34	35.55 ±0.43 (-3.52)	40.23 ±0.76 (+1.17)	49.70 ±0.82	36.90 ±0.34 (-12.80)	41.37 ±0.67 (-8.33)
Qwen3-8B	21.88 ±0.72	22.27 ±0.80 (+0.39)	26.95 ±0.89 (+5.08)	42.26 ±0.67	34.23 ±0.46 (-8.04)	32.74 ±0.67 (-9.52)
LlamaR1-8B	5.08 ±0.40	7.42 ±0.38 (+2.34)	9.77 ±0.66 (+4.69)	10.42 ±0.68	7.44 ±0.37 (-2.98)	10.42 ±0.58 (+0.00)
QwenR1-7B	8.59 ±0.81	6.25 ±0.43 (-2.34)	6.25 ±0.72 (-2.34)	19.64 ±0.74	10.12 ±0.38 (-9.52)	13.69 ±0.69 (-5.95)
	GPQA-DIAMOND*			MMLU-REDUX*		
	DIRECT	1F	2F	DIRECT	1F	2F
GPT-OSS-120B	58.24 ±0.13	35.70 ±0.08 (-22.54)	31.91 ±0.09 (-26.33)	67.59 ±0.02	42.26 ±0.02 (-25.33)	31.61 ±0.03 (-35.97)
GPT-OSS-20B	46.97 ±0.14	16.86 ±0.05 (-30.11)	13.16 ±0.08 (-33.81)	58.86 ±0.04	22.73 ±0.01 (-36.14)	12.83 ±0.02 (-46.03)
Nemotron-32B	66.81 ±0.08	49.43 ±0.10 (-17.38)	37.22 ±0.09 (-29.59)	67.25 ±0.02	32.74 ±0.01 (-34.50)	20.09 ±0.02 (-47.16)
Nemotron-7B	51.61 ±0.11	35.27 ±0.07 (-16.34)	29.40 ±0.07 (-22.21)	56.46 ±0.02	23.58 ±0.02 (-32.88)	14.02 ±0.02 (-42.44)
Qwen3-32B	57.62 ±0.10	34.19 ±0.05 (-23.44)	23.30 ±0.10 (-34.33)	70.62 ±0.02	40.53 ±0.02 (-30.09)	21.97 ±0.02 (-48.65)
Qwen3-8B	47.77 ±0.09	27.13 ±0.08 (-20.64)	20.45 ±0.06 (-27.32)	61.65 ±0.03	27.76 ±0.02 (-33.90)	17.21 ±0.01 (-44.44)
LlamaR1-8B	31.53 ±0.16	20.69 ±0.05 (-10.84)	17.85 ±0.06 (-13.68)	41.17 ±0.03	13.72 ±0.02 (-27.45)	8.89 ±0.02 (-32.28)
QwenR1-7B	21.26 ±0.08	14.73 ±0.09 (-6.53)	15.20 ±0.06 (-6.06)	36.60 ±0.04	13.62 ±0.01 (-22.98)	10.25 ±0.02 (-26.35)
	CRUXEVAL-I*			GAME OF 24*		
	DIRECT	1F	2F	DIRECT	1F	2F
GPT-OSS-120B	89.23 ±0.09	81.02 ±0.05 (-8.21)	77.89 ±0.06 (-11.34)	77.34 ±0.03	56.94 ±0.03 (-20.40)	57.09 ±0.05 (-20.24)
GPT-OSS-20B	61.34 ±0.30	42.39 ±0.11 (-18.95)	40.53 ±0.13 (-20.81)	78.30 ±0.03	41.61 ±0.05 (-36.70)	39.31 ±0.06 (-38.99)
Nemotron-32B	77.06 ±0.09	53.69 ±0.06 (-23.37)	54.19 ±0.09 (-22.87)	79.54 ±0.03	56.68 ±0.03 (-22.86)	51.53 ±0.03 (-28.01)
Nemotron-7B	43.55 ±0.21	35.27 ±0.07 (-8.28)	31.32 ±0.09 (-12.23)	75.35 ±0.02	45.97 ±0.03 (-29.38)	34.77 ±0.02 (-40.58)
Qwen3-32B	75.83 ±0.08	54.89 ±0.04 (-20.94)	50.90 ±0.09 (-24.93)	78.48 ±0.03	43.40 ±0.03 (-35.09)	25.47 ±0.03 (-53.01)
Qwen3-8B	67.49 ±0.08	40.09 ±0.06 (-27.39)	36.20 ±0.08 (-31.28)	76.29 ±0.03	32.92 ±0.02 (-43.38)	23.26 ±0.02 (-53.03)
LlamaR1-8B	30.32 ±0.08	23.84 ±0.07 (-6.48)	21.81 ±0.08 (-8.51)	35.34 ±0.04	5.44 ±0.02 (-29.90)	3.37 ±0.01 (-31.97)
QwenR1-7B	36.84 ±0.11	22.74 ±0.06 (-14.10)	16.82 ±0.06 (-20.01)	40.69 ±0.03	15.60 ±0.02 (-25.09)	8.49 ±0.02 (-32.20)

D Ablation Evaluation of Contextual Drag

D.1 Position and Verification

For the **Position** ablation (POS), we adjust (1) the position of the question to the beginning for more model attention and (2) the draft to the middle of the prompt for less distraction. For the **Verification** ablation (VER), we remove the instruction to explicitly verify drafts.

We provide the implementation details of the **Position** (POS) and **Verification** (VER) ablations below. To enable a fair comparison with the 1F results in Table 1, all ablation prompt templates are adapted from 1F template with minimal changes. We highlight those changes in blue color.

Position Liu et al. (2024) points out that the model tends to make more use of the information at the beginning or end of the input context, which is known as the "lost-in-the-middle" effect. We adjust the position of (1) the question to the beginning and (2) the draft to the middle of the prompt. We expect that this change in position can reduce the use of the in-context draft, therefore alleviating contextual drag.

POS *ablation (*1F*)*

```
In this task, you will be provided with a problem and one draft
solution to that problem.  The problem is as follows:
-- beginning of problem --
{problem}
-- end of problem --

For the draft, walk through the solution step-by-step and check for
any fatal mistakes in the reasoning (for example:  incorrect use of
theorems or facts, incorrect calculations, etc.).  Please note that
once you spot a fatal error in a draft's reasoning, you **MUST** stop
checking the remaining steps of that draft and move on.

For the draft, output an overall verdict on the correctness of the
draft using the following tag format:

<overall_verdict>[Correct / Incorrect]</overall_verdict>

While evaluating individual steps, be strict about algebra, logic,
and correct theorem application.  Here is the draft solution you need
to consider:

### Solution:
-- beginning of the draft --
{draft1}
-- end of the draft --

If the draft is incorrect, write a complete, correct solution (you may
reuse sound ideas from the draft).  If the draft is correct, you may
adopt and polish it as the final solution.

Conclude by presenting the final answer only inside \boxed{} (i.e., the
final answer should appear only inside a LaTeX `\boxed{}`).

Please:
1.  Carefully evaluate the draft solution step-by-step.  Once you spot
a fatal error in a draft's reasoning, you **MUST** stop checking the
remaining steps of that draft and move on.
2.  Output the correctness verdicts for the draft.
3.  Provide a correct solution to the problem with complete reasoning
steps that lead to the answer.

Remember:  conclude with the final answer only in \boxed{}.
```

Verification This ablation investigates the effect of performing an explicit check of the draft before generating the solution. We remove the corresponding instructions in the input prompt so that the model does not need to perform the explicit verification. While an explicit verification can help the model become aware of the mistakes contained in the draft, this also results in more verbose responses and therefore longer contexts that can lead to performance degradation.

VER *ablation (single draft)*

```
In this task, you will be provided with a problem and one draft
solution to that problem.  If the draft is incorrect, write a complete,
correct solution (you may reuse sound ideas from the draft).  If the
draft is correct, you may adopt and polish it as the final solution.

Conclude by presenting the final answer only inside \boxed{} (i.e., the
final answer should appear only inside a LaTeX `\boxed{}`).

The problem is as follows:
-- beginning of problem --
{problem}
-- end of problem --

Here is the draft solution you need to consider:

### Solution:
-- beginning of the draft --
{draft1}
-- end of the draft --

Please provide a correct solution to the problem with complete
reasoning steps that lead to the answer.

Remember:  conclude with the final answer only in \boxed{}.
```

Table 9 shows the results of the **Position** (POS) and **Verification** (VER) ablation evaluation of GPT-OSS-120B and Nemotron-7B/32B, which exhibit more robustness to contextual drag in Table 1, on HMMT25, GPQA, and CRUXEval-I.

Table 9: **Position and Verification ablations**: DIRECT and 1F statistics are copied from Table 1 for comparison. POS and VER denote to the **Position** and **Verification** ablations.

	HMMT25*			
	DIRECT	POS	VER	1F
GPT-OSS-120B	61.01±0.70	53.57±0.77	52.98±0.67	48.51±0.78
Nemotron-32B	80.36±0.62	73.81±0.88	67.56±0.36	65.48±0.78
Nemotron-7B	63.10±0.95	52.38±0.36	51.19±1.30	54.17±0.43
	GPQA-DIAMOND*			
	DIRECT	POS	VER	1F
GPT-OSS-120B	58.24±0.13	42.52±0.09	38.16±0.16	35.70±0.11
Nemotron-32B	66.81±0.08	49.95±0.08	39.44±0.08	49.43±0.14
Nemotron-7B	51.61±0.11	37.59±0.12	27.46±0.05	35.27±0.10
	CRUXEVAL-I*			
	DIRECT	POS	VER	1F
GPT-OSS-120B	89.23±0.08	55.49±0.14	87.13±0.07	81.02±0.06
Nemotron-32B	77.06±0.09	52.76±0.09	52.59±0.13	53.69±0.08
Nemotron-7B	43.55±0.20	32.81±0.13	35.34±0.09	35.27±0.09

D.2 Scaling: Varying the Numbers of In-Context Drafts with Mixed Correctness

We also ablate the number of drafts in context and their correctness and observe the corresponding effects on contextual drag. Specifically, we repeat the same evaluation in Section 2.1 with 1-4 in-context drafts with mixed correctness (1T, 1F; 2T, 1T1F, 2F; 3T, 2T1F, 1T2F, 3F; and 4T, 3T1F, 2T2F, 1T3F, 4F).

As shown in Figure 5, performance degrades sharply when the context contains only incorrect drafts, with more severe drops as the number of incorrect drafts increases. Correct drafts in context generally result in large performance gains, but models can still struggle when incorrect drafts dominate the context (Figure 5), notably for GPT-OSS-120B and Nemotron-32B on CRUXEval-I and GPQA.

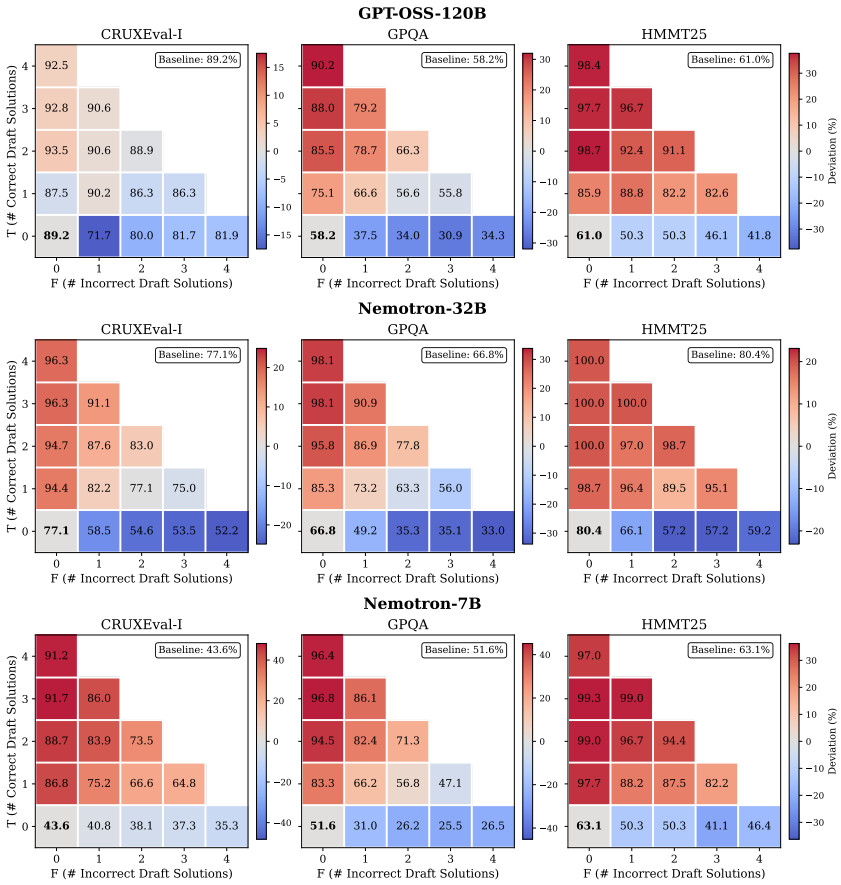


Figure 5: **Scaling ablation**: Performance changes relative to the DIRECT performance as the number of drafts with mixed correctness in context increases.

E TREE-EDIT DISTANCE (TED) ANALYSIS

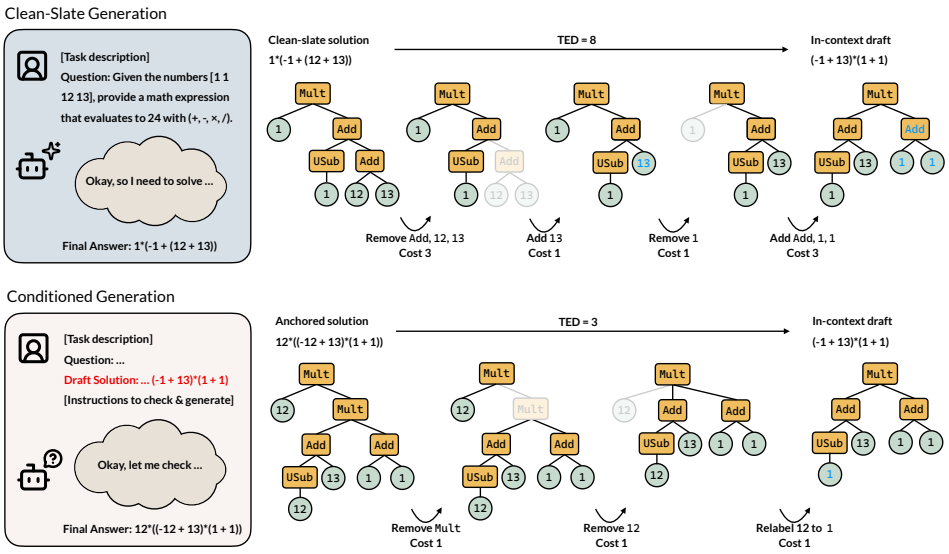


Figure 6: **Overview of TED analysis**: We measure the similarity in reasoning structure of the clean-slate generation, in-context draft, and the conditioned generation using TED. For each pair of clean-slate/conditioned generation and in-context draft, we extract the solution and parse it into a tree, then compute the TED metric between each pair.

E.1 IMPLEMENTATION

We conduct tree-edit-distance (TED) analysis on the synthetic Game of 24 task, which is commonly referred to as a restricted variant of the *Countdown* problem. Since each solution to the Game of 24 puzzle can be parsed into a tree, this allows us to quantify the structural similarity of the reasoning and error patterns underlying the solution.

We take the set of clean-slate generations under DIRECT and the conditioned generations under 1F in Section 2.1. As shown in Figure 6, for each problem, we form two pairs of responses: (1) a clean-slate generation and an in-context draft solution, and (2) a conditioned generation and an in-context draft solution. For each pair, we extract the solutions from the reasoning traces and parse them into trees. Then we compute the TED between each pair.

In the original formulation, the TED metric measures the dissimilarity between two trees by the minimum-cost sequence of edit operations needed to transform one tree into the other Zhang & Shasha (1989). The computation of TED involves three types of operations: (1) node insertion, (2) node deletion, and (3) node relabeling (or substitution). Figure 6 provides an example of TED computation used in our analysis.

E.2 ADDITIONAL ANALYSIS OF CONTEXTUAL DRAG UNDER 2F

We also repeat the same analysis from Section 2.3 for conditioned generation under 2F. Since there are two in-context drafts, we define

$$\text{TED}'(\text{generation}, \{\text{draft1}, \text{draft2}\}) = \min(\text{TED}(\text{generation}, \text{draft1}), \text{TED}(\text{generation}, \text{draft2}))$$

Note that this formulation necessarily results in lower absolute TED values, but does not imply that contextual drag is more severe under 2F than under 1F. Since we define this for both clean-slate and conditioned generation, the comparison of their TED values remains fair. Figure 7 reveals a similar trend to Figure 2b, suggesting that increased diversity of drafts in context does not substantially alleviate contextual drag.

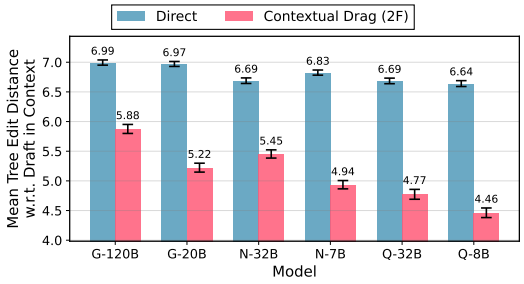


Figure 7: **TED analysis of conditioned generations under 2F**

E.3 Ablation of Answer Reuse Instruction

Another potential confound for the reduced TED distance is the explicit instruction that prompts the model to reuse parts of the solution if it finds them correct. In this ablation setting, we show that removing such reuse instruction does not eliminate the edit distance gap. Below we highlight the removal of ~~the reuse instruction in the prompt~~.

1F *(no reuse instruction)*

```
In this task, you will be provided with a problem and one draft
solution to that problem.  For the draft, walk through the solution
step-by-step and check for any fatal mistakes in the reasoning (for
example:  incorrect use of theorems or facts, incorrect calculations,
etc.).  Please note that once you spot a fatal error in a draft's
reasoning, you **MUST** stop checking the remaining steps of that draft
and move on.

For the draft, output an overall verdict on the correctness of the
draft using the following tag format:

<overall_verdict>[Correct / Incorrect]</overall_verdict>

While evaluating individual steps, be strict about algebra, logic,
and correct theorem application.  If the draft is incorrect, write a
complete, correct solution~~ (you may reuse sound ideas from the draft)~~.
If the draft is correct, you may adopt and polish it as the final
solution.

Conclude by presenting the final answer only inside \boxed{} (i.e., the
final answer should appear only inside a LaTeX `\boxed{}`).

The problem is as follows:
-- beginning of problem --
{problem}
-- end of problem --

Here is the draft solution you need to consider:

### Solution:
-- beginning of the draft --
{draft1}
-- end of the draft --

Please:
1.  Carefully evaluate the draft solution step-by-step.  Once you spot
a fatal error in a draft's reasoning, you **MUST** stop checking the
remaining steps of that draft and move on.
2.  Output the correctness verdicts for the draft.
3.  Provide a correct solution to the problem with complete reasoning
steps that lead to the answer.

Remember:  conclude with the final answer only in \boxed{}.
```

As shown in Figure 8, removing the reuse instruction does not significantly change the trend, which remains consistent with Figure 2b and Figure 3b. This further supports our interpretation that the reduced TED distance is driven by contextual drag rather than instruction-following behavior.

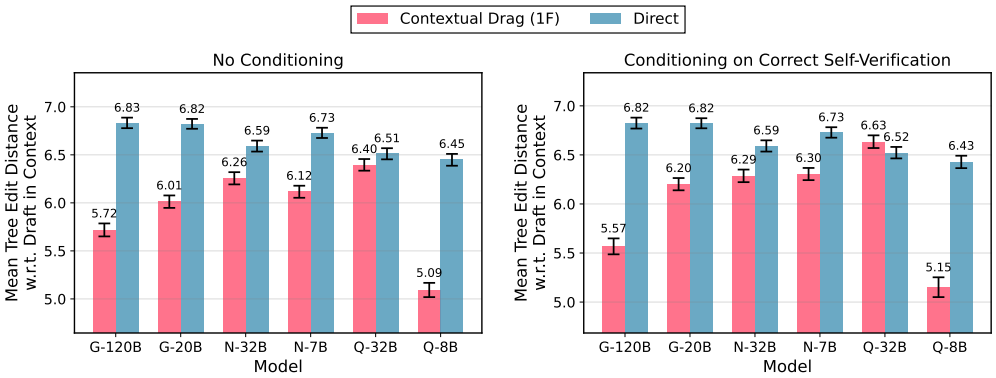


Figure 8: **TED analysis of conditioned generations under 1F without reuse instruction in the prompt**

F ADDITIONAL RESULTS OF CONTEXTUAL DRAG DESPITE ERROR SIGNALS

External Error Signal Here we provide the modified prompt used in Section 3.1 which explicitly and repeatedly stated the incorrectness of the draft in context.

External Error Signal (1F)

```
In this task, you will be provided with a problem and one draft
solution with **INCORRECT** final answer to that problem.  For the
**INCORRECT** draft, walk through the solution step-by-step and check
for any fatal mistakes in the reasoning (for example:  incorrect use
of theorems or facts, incorrect calculations, etc.).  Please note that
once you spot a fatal error in a draft's reasoning, you **MUST** stop
checking the remaining steps of that draft and move on.

For the **INCORRECT** draft, output an overall verdict on the
correctness of the draft using the following tag format:

<overall_verdict>Incorrect</overall_verdict>

While evaluating individual steps, be strict about algebra, logic,
and correct theorem application.  Finally, write a complete, correct
solution (you may reuse sound ideas from the draft, but **DO NOT COPY**
the final answer because it is **INCORRECT**).

Conclude by presenting the final answer only inside \boxed{} (i.e., the
final answer should appear only inside a LaTeX `\boxed{}`).

The problem is as follows:
-- beginning of problem --
{problem}
-- end of problem --

Here is the draft solution with **INCORRECT** final answer you need to
consider:

### Solution:
-- beginning of the draft --
{draft1}
-- end of the draft --

Please:
1.  Carefully evaluate the draft solution step-by-step.  Once you spot
a fatal error in a draft's reasoning, you **MUST** stop checking the
remaining steps of that draft and move on.
2.  Output the correctness verdicts for the draft.
3.  Provide a correct solution to the problem with complete reasoning
steps that lead to the answer.

Remember:  the draft solution has **INCORRECT** final answer.  Please
conclude with the final answer only in \boxed{}.
```

Table 10: Full results for Figure 3a, where E-Error refers to the performance under external error signal.

Model	AIME24*		AIME25*		HMMT24*		HMMT25*	
	DIRECT	E-ERROR	DIRECT	E-ERROR	DIRECT	E-ERROR	DIRECT	E-ERROR
GPT-OSS-120B	66.2	25.0 (-41.2)	66.8	34.1 (-32.7)	48.8	26.2 (-22.7)	61.0	40.0 (-21.0)
GPT-OSS-20B	51.9	17.5 (-34.4)	51.9	17.7 (-34.2)	40.2	13.3 (-27.0)	58.6	27.1 (-31.5)
Nemotron-32B	83.8	76.9 (-6.9)	78.4	71.4 (-7.0)	61.7	57.4 (-4.3)	80.4	77.1 (-3.3)
Nemotron-7B	67.5	60.6 (-6.9)	67.8	56.2 (-11.5)	49.2	43.8 (-5.5)	63.1	59.2 (-3.9)
Qwen3-32B	65.0	48.8 (-16.2)	54.8	46.9 (-7.9)	39.1	37.9 (-1.2)	49.7	42.9 (-6.8)
Qwen3-8B	55.6	41.9 (-13.8)	45.7	30.8 (-14.9)	21.9	27.3 (+5.5)	42.3	33.9 (-8.3)
Model	GPQA-DIAMOND*		MMLU-REDUX*		CRUXEVAL-I*		GAME OF 24*	
	DIRECT	E-ERROR	DIRECT	E-ERROR	DIRECT	E-ERROR	DIRECT	E-ERROR
GPT-OSS-120B	58.2	44.1 (-14.1)	67.6	73.8 (+6.2)	89.2	80.6 (-8.6)	77.3	62.6 (-14.7)
GPT-OSS-20B	47.0	32.1 (-14.8)	58.9	50.8 (-8.1)	61.3	48.7 (-12.6)	78.3	74.4 (-3.9)
Nemotron-32B	66.8	64.9 (-1.9)	67.2	80.3 (+13.0)	77.1	68.6 (-8.5)	79.5	74.1 (-5.5)
Nemotron-7B	51.6	50.3 (-1.4)	56.5	64.2 (+7.7)	43.6	39.4 (-4.2)	75.4	69.4 (-6.0)
Qwen3-32B	57.6	54.9 (-2.7)	70.6	80.5 (+9.9)	75.8	67.2 (-8.6)	78.5	60.5 (-17.9)
Qwen3-8B	47.8	38.2 (-9.6)	61.7	66.1 (+4.4)	67.5	51.6 (-15.9)	76.3	48.9 (-27.4)

Table 11: Full results for Figure 3b, where SD-Error refers to the performance under self-detected error signal. We also show the number of problems remaining after the post-hoc filtering next to the direct performance.

Model	AIME24*		AIME25*		HMMT24*		HMMT25*	
	DIRECT	SD-ERROR	DIRECT	SD-ERROR	DIRECT	SD-ERROR	DIRECT	SD-ERROR
GPT-OSS-120B	66.2 (10/10)	53.6 (-12.7)	66.8 (13/13)	70.9 (+4.0)	48.8 (16/16)	52.9 (+4.1)	60.5 (19/21)	64.3 (+3.7)
GPT-OSS-20B	51.9 (10/10)	28.5 (-23.4)	51.9 (13/13)	43.3 (-8.6)	40.2 (16/16)	16.4 (-23.8)	57.2 (20/21)	33.9 (-23.3)
Nemotron-32B	83.8 (10/10)	93.3 (+9.5)	78.4 (13/13)	80.6 (+2.3)	61.7 (16/16)	66.2 (+4.5)	80.4 (21/21)	85.7 (+5.3)
Nemotron-7B	67.5 (10/10)	76.0 (+8.5)	67.8 (13/13)	83.1 (+15.3)	50.8 (15/16)	54.4 (+3.6)	63.1 (21/21)	72.1 (+9.0)
Qwen3-32B	65.0 (10/10)	49.0 (-16.0)	55.7 (12/13)	53.7 (-2.1)	39.1 (16/16)	47.3 (+8.2)	50.3 (20/21)	51.4 (+1.1)
Qwen3-8B	61.8 (9/10)	54.1 (-7.7)	50.0 (8/13)	44.3 (-5.7)	20.1 (14/16)	36.2 (+16.1)	46.5 (18/21)	46.3 (-0.2)
Model	GPQA-DIAMOND*		MMLU-REDUX*		CRUXEVAL-I*		GAME OF 24*	
	DIRECT	SD-ERROR	DIRECT	SD-ERROR	DIRECT	SD-ERROR	DIRECT	SD-ERROR
GPT-OSS-120B	57.9 (112/132)	48.9 (-9.1)	73.0 (450/569)	79.7 (+6.7)	90.6 (173/188)	85.7 (-4.9)	77.5 (634/653)	80.8 (+3.4)
GPT-OSS-20B	44.8 (115/132)	25.0 (-19.8)	61.4 (458/569)	61.3 (-0.1)	61.5 (175/188)	53.5 (-8.0)	78.5 (603/653)	79.0 (+0.5)
Nemotron-32B	66.5 (104/132)	66.0 (-0.5)	77.1 (293/569)	74.2 (-2.8)	79.1 (164/188)	74.7 (-4.4)	79.4 (649/653)	76.0 (-3.4)
Nemotron-7B	50.8 (103/132)	56.5 (+5.7)	62.7 (266/569)	59.9 (-2.8)	44.4 (163/188)	52.4 (+8.0)	75.6 (641/653)	76.8 (+1.3)
Qwen3-32B	55.3 (110/132)	43.3 (-12.1)	74.6 (426/569)	77.4 (+2.9)	75.5 (161/188)	72.5 (-3.0)	78.3 (645/653)	70.3 (-8.0)
Qwen3-8B	44.5 (104/132)	32.3 (-12.2)	70.4 (332/569)	67.9 (-2.4)	67.3 (158/188)	60.1 (-7.2)	76.2 (576/653)	74.7 (-1.5)

G Additional Details of Context Manipulation Experiments

G.1 Implementation of Revise and Filter

We provide the full implementation details of both methods below. We note that when using the output from a previous step, we always remove the thinking traces wrapped in `<think></think>` for manageable context length. We only present the prompt templates for 1F for illustration.

Revise consists of 2 steps of inference:

(revise): given the **`problem`** and one **`draft`**, the model is asked to fix any mistakes and produce a revised solution. Denote the output as the **`revised_draft`**;

Revise *(revise step, 1F)*

```
You are given a problem and one draft solution to that problem.  Your
task is to revise the draft to produce an improved, complete solution
that addresses any mistakes or shortcomings in the original draft.
Ensure that your revised solution includes all necessary reasoning
steps leading to the final answer and conclude by presenting the final
answer only inside \boxed{}.

The problem is as follows:
-- beginning of problem --
{problem}
-- end of problem --

Here is the draft solution you need to revise:
-- beginning of the draft --
{draft1}
-- end of the draft --

Remember:  conclude with the final answer only in \boxed{}.
```

(solve): given the **`problem`** and the **`revised_draft`**, the model generates a solution. The template is the same as the VER ablation template except that the draft is replaced with its revised version. We remove the instruction to check the draft correctness to optimize model performance.

Revise *(solve step, 1F)*

```
In this task, you will be provided with a problem and one draft
solution to that problem.  If the draft is incorrect, write a complete,
correct solution (you may reuse sound ideas from the draft).  If the
draft is correct, you may adopt and polish it as the final solution.

Conclude by presenting the final answer only inside \boxed{} (i.e., the
final answer should appear only inside a LaTeX `\boxed{}`).

The problem is as follows:
-- beginning of problem --
{problem}
-- end of problem --

Here is the draft solution you need to consider:

### Solution:
-- beginning of the draft --
{revised_draft1}
-- end of the draft --

Please provide a correct solution to the problem with complete
reasoning steps that lead to the answer.

Remember:  conclude with the final answer only in \boxed{}.
```

FILTER consists of 3 steps of inference:

(strategy): given the **problem**, the model produces a high-level strategy to solve the problem without including any intermediate steps or the final answer. Denote the output as the **strategy**;

FILTER *(strategy stage, single draft)*

```
You are given a problem.  Your task is **NOT to solve the problem**.
Instead, produce a **high-level strategy or a plan** that you would use
to approach the problem.

Requirements for your response:
1.  **Do NOT give any step-by-step computation or any final answers**.
2.  Produce a concise, structured plan (use numbered steps or short
bullet points) of you would take as if you would attempt to solve the
given question.  You can include any useful heuristics, ways to reduce
problem complexity, and common pitfalls to avoid.
3.  Finally, include a short checklist to verify a candidate answer.

Now, produce the high-level strategy (no solution) for the following
problem:
-- beginning of problem --
{problem}
-- end of problem --

Remember that you **do not attempt to solve the question or include any
intermediate steps or the final answer in your response**.
```

(filter): given the **problem**, the **strategy**, and one **draft**, the model is asked to filter the draft to only keep the correct and useful steps toward solving the problem according to the **strategy** it produced in the previous step. Similar to the revise step in REVISE, this step needs to be repeated for each draft independently. Denote the output as the **filtered_draft**;

FILTER *(filter step,* 1F*)*

```
You are given a problem, a draft solution attempt, and a high-level
strategy for solving the problem.  Your task is to identify the useful
components or intermediate steps in the draft solution that align
with the given strategy.  Focus on selecting steps that contribute
meaningfully to progressing toward the final solution, even if the
draft solution as a whole may contain mistakes or irrelevant parts.
Format your response as numbered steps, followed by a concise (no more
than 2 sentences), clear explanation of how each step fits into the
strategy.  Remember:  **only include the useful and correct parts** and
**do not include any incorrect or irrelevant parts** in your response.

The problem is as following:
-- beginning of problem --
{problem}
-- end of problem --

You should consider the following strategy to solve the problem:
-- beginning of strategy --
{strategy}
-- end of strategy --

Please **only output the useful and correct components or intermediate
steps that align with the above strategy** from the following draft
solution:
-- beginning of draft solution --
{draft1}
-- end of draft solution --

Remember to format your response as numbered steps, followed by a
concise explanation of why they are useful.
```

(solve): given the **problem** and the **filtered_draft**, the model is asked to generate a new solution. Similar to REVISE, the prompt template used at this step is the same as the 1F template in Section C.1 except that (1) the draft in the context is replaced with its revised version and (2) the instruction to check the correctness of the draft is removed to optimize model performance.

FILTER *(solve step,* 1F*)*

```
In this task, you will be provided with a problem and one draft
solution to that problem.  If the draft is incorrect, write a complete,
correct solution (you may reuse sound ideas from the draft).  If the
draft is correct, you may adopt and polish it as the final solution.

Conclude by presenting the final answer only inside \boxed{} (i.e., the
final answer should appear only inside a LaTeX `\boxed{}`).

The problem is as follows:
-- beginning of problem --
{problem}
-- end of problem --

Here is the draft solution you need to consider:

### Solution:
-- beginning of the draft --
{filtered_draft1}
-- end of the draft --

Please provide a correct solution to the problem with complete
reasoning steps that lead to the answer.

Remember:  conclude with the final answer only in \boxed{}.
```

G.2 Additional Results

Here we provide the full results of the two context denoising approaches discussed in Section 4.1. Results for 1F are shown in Table 12 and results for 2F are shown in Table 13.

Table 12: Additional results of context denoising experiments (1F) with 95% confidence internal.

	AIME24*					AIME25*				
	Direct	Revise	Filter	1F	1F-VER	Direct	Revise	Filter	1F	1F-VER
GPT-OSS-120B	66.25±1.72	80.00±0.55	63.75±2.10	43.75±0.73	57.50±0.95	66.83±1.47	68.27±1.03	65.38±1.52	55.29±1.47	58.17±1.50
GPT-OSS-20B	51.88±1.47	52.50±0.77	53.12±1.21	17.50±2.26	41.25±1.80	51.92±1.54	48.56±1.43	47.60±1.22	18.75±1.20	46.63±2.54
Nemotron-32B	83.75±1.80	74.37±1.02	84.38±1.02	67.50±1.82	66.88±1.33	78.37±1.18	73.56±1.07	84.13±0.58	61.54±1.28	62.50±1.21
Nemotron-7B	67.50±1.28	63.13±1.33	68.12±0.94	53.12±1.73	56.25±2.24	67.79±1.12	63.94±0.73	70.19±0.72	52.40±0.97	46.63±0.78
Qwen3-32B	65.00±2.05	46.25±1.31	65.62±1.28	41.25±1.20	43.75±3.04	54.81±1.47	37.98±0.78	53.85±0.98	31.25±0.94	34.13±1.54
Qwen3-8B	55.62±2.12	50.00±1.23	66.25±1.20	30.63±1.28	31.87±1.21	45.67±1.20	39.42±0.96	52.88±1.03	20.19±1.09	19.23±1.23
	HMMT24*					HMMT25*				
	Direct	Revise	Filter	1F	1F-VER	Direct	Revise	Filter	1F	1F-VER
GPT-OSS-120B	48.83±1.22	57.03±0.59	44.92±1.18	39.84±0.84	43.36±1.56	61.01±0.70	68.75±0.52	59.23±0.94	48.51±0.78	52.98±0.67
GPT-OSS-20B	40.23±1.08	50.00±0.47	30.86±0.42	13.28±1.17	30.08±0.98	58.63±0.78	62.20±0.68	45.83±0.50	20.54±0.97	42.26±0.53
Nemotron-32B	61.72±1.40	62.11±0.99	66.02±0.83	51.95±0.76	50.78±1.00	80.36±0.62	74.40±0.56	91.96±0.47	65.48±0.78	67.56±0.36
Nemotron-7B	49.22±1.15	49.61±0.92	43.75±0.76	35.94±0.94	35.55±1.32	63.10±0.95	63.10±0.47	62.20±0.73	54.17±0.43	51.19±1.30
Qwen3-32B	39.06±1.34	41.02±0.57	34.77±0.78	35.55±0.60	29.69±0.54	49.70±0.83	48.81±0.44	43.15±0.80	36.90±0.47	40.77±0.77
Qwen3-8B	21.88±0.72	30.47±0.52	26.56±0.54	22.27±1.13	23.05±0.85	42.26±0.67	38.69±0.62	42.86±0.51	34.23±0.65	36.61±0.72
	GPQA-Diamond*					MMLU-Redux*				
	Direct	Revise	Filter	1F	1F-VER	Direct	Revise	Filter	1F	1F-VER
GPT-OSS-120B	58.24±0.13	46.88±0.09	54.88±0.14	35.70±0.11	38.16±0.16	67.59±0.02	48.11±0.02	63.70±0.03	42.26±0.03	40.49±0.03
GPT-OSS-20B	46.97±0.14	33.33±0.07	43.70±0.16	16.86±0.07	26.66±0.12	58.86±0.04	35.54±0.03	55.26±0.03	22.73±0.01	25.48±0.04
Nemotron-32B	66.81±0.08	48.15±0.06	68.47±0.07	49.43±0.14	39.44±0.08	67.25±0.02	31.38±0.01	64.54±0.01	32.74±0.01	24.75±0.02
Nemotron-7B	51.61±0.11	35.27±0.07	55.73±0.11	35.27±0.10	27.46±0.05	56.46±0.02	19.63±0.01	51.58±0.02	23.58±0.03	15.99±0.02
Qwen3-32B	57.62±0.10	27.04±0.05	58.38±0.06	34.19±0.07	22.68±0.08	70.62±0.02	30.23±0.01	39.08±0.02	40.53±0.03	23.91±0.01
Qwen3-8B	47.77±0.09	25.24±0.05	45.27±0.06	27.13±0.11	18.75±0.06	61.65±0.03	28.77±0.01	57.96±0.02	27.76±0.02	20.05±0.03
	CRUXEval-I*					Game of 24*				
	Direct	Revise	Filter	1F	1F-VER	Direct	Revise	Filter	1F	1F-VER
GPT-OSS-120B	89.23±0.08	81.98±0.05	87.80±0.06	81.02±0.06	87.13±0.07	77.34±0.03	73.19±0.02	78.91±0.01	56.94±0.05	57.20±0.06
GPT-OSS-20B	61.34±0.28	61.24±0.10	49.73±0.14	42.39±0.14	45.11±0.12	78.30±0.03	56.93±0.05	75.20±0.08	41.61±0.07	39.76±0.07
Nemotron-32B	77.06±0.09	67.49±0.06	73.80±0.06	53.69±0.08	52.59±0.13	79.54±0.03	74.62±0.01	82.30±0.01	56.68±0.04	51.00±0.02
Nemotron-7B	43.55±0.20	48.34±0.06	39.13±0.08	35.27±0.09	35.34±0.09	75.35±0.02	71.18±0.02	73.96±0.02	45.97±0.04	38.97±0.03
Qwen3-32B	75.83±0.08	65.36±0.03	62.90±0.06	54.89±0.06	53.96±0.04	78.48±0.03	58.26±0.02	79.39±0.03	43.40±0.05	41.62±0.03
Qwen3-8B	67.49±0.07	49.00±0.06	51.80±0.09	40.09±0.08	38.33±0.04	76.29±0.03	42.71±0.02	76.67±0.02	32.92±0.03	29.34±0.02

Table 13: Additional results of context denoising experiments (2F) with 95% confidence internal.

	AIME24*					AIME25*				
	DIRECT	REVISE	FILTER	2F	2F-VER	DIRECT	REVISE	FILTER	2F	2F-VER
GPT-OSS-120B	66.25±1.72	61.87±1.25	57.50±1.60	43.75±2.04	55.00±2.19	66.83±1.47	68.27±0.73	54.33±1.45	40.38±1.30	56.73±1.52
GPT-OSS-20B	51.88±1.47	47.50±1.16	49.38±2.01	21.25±2.32	40.62±2.01	51.92±1.54	54.33±1.01	50.48±1.22	20.67±1.51	46.15±1.11
Nemotron-32B	83.75±1.80	70.62±1.02	83.12±1.31	63.13±1.42	60.62±1.39	78.37±1.18	71.63±0.80	71.63±0.71	50.96±1.47	63.46±1.26
Nemotron-7B	67.50±1.28	64.38±0.77	71.25±1.21	58.75±1.80	51.88±1.25	67.79±1.12	48.56±0.71	75.96±0.97	37.02±1.29	41.83±1.28
Qwen3-32B	65.00±2.05	35.62±1.34	60.62±1.02	30.00±1.55	47.50±1.69	54.81±1.47	38.46±0.98	58.65±0.73	31.25±0.78	30.29±1.01
Qwen3-8B	55.62±2.12	34.38±0.94	65.62±1.22	20.62±1.69	24.38±1.22	45.67±1.20	15.38±0.64	43.27±0.97	11.06±1.04	15.38±1.33
	HMMT24*					HMMT25*				
	DIRECT	REVISE	FILTER	2F	2F-VER	DIRECT	REVISE	FILTER	2F	2F-VER
GPT-OSS-120B	48.83±1.22	59.38±0.38	48.05±1.14	45.31±1.06	45.31±0.99	61.01±0.70	59.82±0.51	57.44±0.66	49.11±0.69	55.36±0.82
GPT-OSS-20B	40.23±1.08	57.03±1.11	50.39±1.10	17.97±1.21	35.94±0.88	58.63±0.78	56.85±0.66	52.68±0.77	24.70±1.02	44.05±0.89
Nemotron-32B	61.72±1.40	59.77±0.47	67.97±1.01	50.78±0.85	47.66±0.89	80.36±0.62	76.19±0.44	79.46±0.50	64.58±0.70	62.50±0.69
Nemotron-7B	49.22±1.15	56.64±0.99	57.42±0.87	40.62±0.76	32.42±0.95	63.10±0.95	68.45±0.30	75.30±0.45	52.38±0.74	49.11±0.78
Qwen3-32B	39.06±1.34	29.30±0.75	33.20±0.80	40.23±0.76	26.17±0.82	49.70±0.83	47.02±0.50	50.60±0.62	41.37±0.67	34.23±0.81
Qwen3-8B	21.88±0.72	26.95±0.45	25.78±0.46	26.95±0.89	18.36±0.96	42.26±0.67	40.48±0.44	44.35±0.78	32.74±0.67	32.74±0.72
	GPQA-DIAMOND*					MMLU-REDUX*				
	DIRECT	REVISE	FILTER	2F	2F-VER	DIRECT	REVISE	FILTER	2F	2F-VER
GPT-OSS-120B	58.24±0.13	43.56±0.11	53.17±0.15	31.91±0.09	31.20±0.08	67.59±0.02	46.67±0.02	63.43±0.02	31.61±0.03	30.12±0.02
GPT-OSS-20B	46.97±0.14	30.87±0.12	42.28±0.18	13.16±0.08	20.27±0.14	58.86±0.04	35.19±0.01	53.71±0.02	12.83±0.02	14.95±0.03
Nemotron-32B	66.81±0.08	46.64±0.05	63.87±0.05	37.22±0.09	34.04±0.09	67.25±0.02	30.71±0.01	67.43±0.01	20.09±0.02	17.03±0.02
Nemotron-7B	51.61±0.11	35.80±0.05	49.72±0.08	29.40±0.07	27.56±0.06	56.46±0.02	21.10±0.01	51.56±0.01	14.02±0.02	11.63±0.02
Qwen3-32B	57.62±0.10	26.09±0.03	31.72±0.07	23.30±0.10	20.08±0.08	70.62±0.02	26.58±0.01	66.93±0.02	21.97±0.02	14.69±0.01
Qwen3-8B	47.77±0.09	26.42±0.03	46.59±0.06	20.45±0.06	17.99±0.06	61.65±0.03	28.15±0.01	57.71±0.02	17.21±0.01	12.63±0.01
	CRUXEVAL-I*					GAME OF 24*				
	DIRECT	REVISE	FILTER	2F	2F-VER	DIRECT	REVISE	FILTER	2F	2F-VER
GPT-OSS-120B	89.23±0.08	75.96±0.09	79.06±0.12	77.89±0.06	83.54±0.09	77.34±0.03	72.27±0.02	78.81±0.03	57.09±0.05	57.78±0.04
GPT-OSS-20B	61.34±0.28	61.90±0.13	44.68±0.18	40.53±0.12	44.18±0.12	78.30±0.03	56.07±0.04	76.37±0.05	39.31±0.06	40.02±0.07
Nemotron-32B	77.06±0.09	72.61±0.07	71.34±0.06	54.19±0.09	51.80±0.06	79.54±0.03	74.41±0.02	81.35±0.02	51.53±0.03	50.63±0.03
Nemotron-7B	43.55±0.20	47.91±0.07	38.36±0.06	31.32±0.09	36.10±0.12	75.35±0.02	70.55±0.01	73.62±0.02	34.77±0.02	39.11±0.04
Qwen3-32B	75.83±0.08	69.32±0.04	62.23±0.07	50.90±0.08	51.66±0.09	78.48±0.03	56.99±0.02	78.18±0.03	25.47±0.03	41.54±0.03
Qwen3-8B	67.49±0.07	53.86±0.04	49.80±0.06	36.20±0.08	36.20±0.06	76.29±0.03	43.78±0.01	76.21±0.01	23.26±0.02	29.24±0.03

H Additional Information for Supervised Fine-tuning Experiments

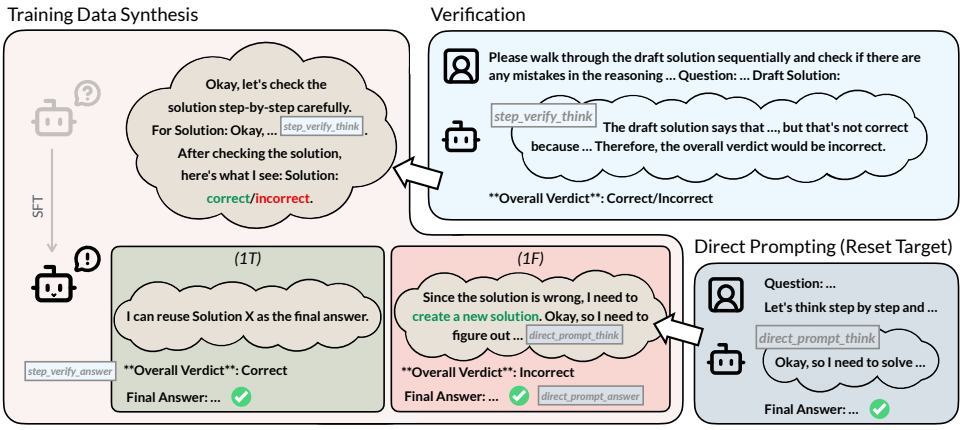


Figure 9: **Overview of the proposed training procedure**: Synthesized reasoning traces teach the model to verify in-context drafts before solving. If the model identifies the draft as incorrect, as in the case of 1F shown above, it is trained to ignore the revert to a reasoning trajectory samples from clean-slate generation. If the model verified the draft to be correct, it is trained to reuse the answer in draft.

In this section, we provide more details on the data curation and training setup for the supervised fine-tuning experiment described in Section 4.2.

H.1 Question Selection from BigMathRL

To obtain a set of questions on which we can achieve reliable verification, we draw problems from BigMathRL Albalak et al. (2025) and restrict to a reasonably hard subset using an external solver filter: we retain only questions whose solve rate under Llama3.1-8B (provided in the dataset) is below 0.5. From this filtered pool, we subsample $40{,}000$ questions with a fixed source mixture across the underlying BigMathRL data sources.

We report the data composition of the question pool in Table 14. We conducted balanced subsampling such that for small sources, we include the full available set.

Table 14: Source mixture of the 40k question subset used for synthesis. "Available" denotes the number of filtered questions per source after applying the Llama 3.1 8B solve-rate < 0.5 criterion.

Source	Available	Sampled	Sample rate
cn_k12	37,228	6,729	18.1%
orca_math	27,153	6,728	24.8%
olympiads	29,941	6,728	22.5%
aops_forum	5,067	5,067	100.0%
gsm8k	159	159	100.0%
amc_aime	71	71	100.0%
math	2,780	2,780	100.0%
omnimath	2,033	2,033	100.0%
openmath	281	281	100.0%
harp	2,696	2,696	100.0%
big_math	27,094	6,728	24.8%
Total	–	40,000	–

H.2 DRAFT GENERATION AND VERIFIABLE CORRECTNESS

For each selected question, we sample candidate draft solutions from a family of open-weight reasoning models: GPT-OSS-20B (with medium thinking effort), Qwen3-8B (with thinking), OpenReasoning-Nemotron-7B, LlamaR1-8B, and QwenR1-7B. Concretely, we sample 4 independent DIRECT responses per (question, model) pair, yielding up to 20 drafts per question. We then determine draft correctness using the `math_verify` package Kydlíček (2024), which provides verifiable answer checking for math problems.

H.3 VERIFIER TRAJECTORY CURATION

To obtain supervision for verification behavior, we prompt GPT-OSS-20B (our target model for training) to verify each draft solution with the following prompt template, producing a verdict and accompanying verification rationale.

Verification Prompt: Training Data Curation

```
In this task you will be given a question and a draft solution to that
question including the thinking steps.  Please walk through the draft
solution sequentially and check if there are any fatal mistakes in
the reasoning (e.g.  incorrect use of theorems / knowledge, incorrect
calculations, etc.).  Please note that once you spot an fatal error
reasoning process, you **MUST** stop checking the remaining steps
and directly output the verdict.  **DO NOT ATTEMPT TO RESOLVE THE
QUESTION**.  If you can find a potential fix to the fatal step, you
can also output it, but you **MUST NOT** attempt to further resolve
the question.  If all steps appears to be correct, you should check
the entire draft.  In the end you must output an overall verdict to the
correctness of the provided draft in the form of:

<overall_verdict>[Correct / Incorrect]</overall_verdict>
<overall_confidence>[High / Medium / Low]</overall_confidence>
<fix>[Any potential fix to the incorrect step if there is any]</fix>

Please be strict with algebra, logic, and theorem use.  Please only
verify and provide feedback to the provided draft.  You **MUST NOT**
attempt to re-solving the problem.

The question provided is:
-- beginning of question --
{problem}
-- end of question --

Please follow the instructions above and examine the draft solution:
-- beginning of draft --
{draft}
-- end of draft --

Please note that once you spot a fatal error reasoning process, you
**MUST** stop checking the remaining steps and directly output the
verdict.  If all steps appears to be correct, you should check the
entire draft.  **DO NOT ATTEMPT TO RESOLVE THE QUESTION**
```

The prompt above is designed such that we can accurately parse the model's final verification verdict wrapped between `<overall_verdict>` and `<\overall_verdict>`.

We compare the model's verification verdict with the symbolic verifier's result and retain only verification trajectories that agree with the verifier outcome. This produces a set of correct verification trajectories for both correct and incorrect drafts.

H.4 BALANCED TRAINING SET CONSTRUCTION

We construct the SFT dataset by sampling: (i) $40{,}000$ (question, correct draft, correct verification) triples and (ii) $40{,}000$ (question, incorrect draft, correct verification) triples. This yields $80{,}000$ total training examples with a balanced mixture of correct / incorrect drafts. Each example, depending on whether it contains a correct or incorrect draft, is formatted using a dedicated template (as shown below). `verification_reasoning_trace` is the reasoning trace collected in Section H.3 under the verification prompt, and `clean_slate_reasoning_trace` is the reasoning trace generated by GPT-OSS-20B when correctly solving the problem under the DIRECT setting with no additional context.

Synthetic Reasoning Trace: Correct Draft

```
<|channel|>analysis<|message|>We need to check the draft.  If it is
correct, I can reuse it as the final answer.  If it is wrong, I need
to come up with a correct solution myself, reusing any useful pieces if
possible.  {verification_reasoning_trace}

We can see that the draft is Correct.  I can reuse the draft solution
as the final answer.
<|end|><|start|>assistant<|channel|>final<|message|><overall_verdict_1>
Correct</overall_verdict_1>

The final answer is:

{final_answer}
```

Synthetic Reasoning Trace: Incorrect Draft

```
<|channel|>analysis<|message|>We need to check the draft.  If it is
correct, I can reuse it as the final answer.  If it is wrong, I need
to come up with a correct solution myself, reusing any useful pieces if
possible.  {verification_reasoning_trace}

We can see that the draft is Incorrect.  Since the draft is not
correct, I need to create a new solution.  I need to think about this
problem again.

{clean_slate_reasoning_trace}

<|end|><|start|>assistant<|channel|>final<|message|><overall_verdict_1>
Incorrect</overall_verdict_1>

The final answer is:

{final_answer}
```

H.5 TRAINING SETUP

We perform full supervised fine-tuning for one epoch on the synthesized dataset using cosine learning rate decay with a peak learning rate of 5×10^{-5} and a 10% warmup fraction. Training was conducted on $8\times$H100 nodes with Llama-factory Zheng et al. (2024).

H.6 ADDITIONAL RESULTS: 1T EVALUATION AFTER SFT

In Section 4.2 we fine-tune GPT-OSS-20B to (i) produce accurate verification trajectories and (ii) "reset" to its DIRECT reasoning trace when verification indicates that the in-context draft is incorrect. Here we report additional evaluation results on the 1T setting (when there is a correct draft in context) after training. We evaluate the trained model using the same set of questions and draft-construction pipeline as in our main 1T and DIRECT evaluations, with the only exception being that the in-context draft is sampled to be a correct, as determined by the symbolic verifier.

Table 15: **Targeted SFT decreases 1T performance.** Pass@1 accuracy on competitive math benchmarks under 1T (verify-then-answer) for GPT-OSS-20B baseline and our finetuned checkpoint. Deltas in parentheses are absolute percentage-point changes vs. baseline.

SETTING	AIME24*	AIME25*	HMMT24*	HMMT25*
1T (PASS@1)				
Baseline	73.12	73.56	70.31	76.79
Finetuned	50.63 (-22.49)	49.04 (-24.52)	44.92 (-25.39)	49.11 (-27.68)
1T (PASS@5)				
Baseline	100.00	94.71	95.31	97.02
Finetuned	83.75 (-16.25)	90.87 (-3.84)	85.94 (-9.37)	85.42 (-11.60)

Table 15 shows that while our targeted SFT improves robustness in the 1F erroneous-draft setting (main text), it substantially reduces 1T pass@1 and pass@5 performance across benchmarks. A plausible explanation is a *robustness-utilization tradeoff*: training the model to "reset" after error detection may also make it less able to capitalize on useful information in the provided draft, even when that context is partially or fully correct. In 1T, this can manifest as overly conservative behavior (discarding potentially helpful structure) or increased token/attention budget spent on verification-style trajectories rather than constructive problem solving, leading to lower pass rates. Together, these results suggest targeted SFT is a proof-of-concept mitigation that improves robustness to incorrect context but is not yet a complete intervention for settings where correct context utilization is important.

H.7 TRAINING WITH GRPO

In addition to the curated supervised fine-tuning (SFT) recipe in Section 4.2, we also experimented with standard Group Relative Policy Optimization (GRPO; **?**) as a post-training alternative. We train on the *same* set of math problems used for SFT, but optimize only a final-answer correctness reward.

Concretely, for each prompt we sample a group of $n{=}8$ rollouts from the current policy, compute a binary reward based on whether the final answer matches the ground truth, and apply the GRPO update. We use batchsize 64, maximum context length 16,384, and otherwise follow standard GRPO defaults (e.g., on-policy sampling, per-step KL regularization to the reference policy, and reward normalization within each group).

Table 16: **GRPO results on competitive math under contextual drag (1F).** We report accuracy (%) in the 1F setting. "Finetuned" denotes our targeted SFT model (Section 4.2); "GRPO" denotes the model trained with standard GRPO. Deltas in parentheses are absolute percentage-point changes relative to the BASELINE in the same block.

SETTING	AIME24*	AIME25*	HMMT24*	HMMT25*
1F				
Baseline	17.5	18.8	13.3	21.2
Finetuned	40.6 (+23.1)	20.7 (+1.9)	19.2 (+5.9)	27.8 (+6.6)
GRPO	22.5 (+5.0)	20.7 (+1.9)	17.6 (+4.3)	24.4 (+3.2)
1F (CONDITIONED ON SELF-DETECTED ERROR SIGNAL)				
Baseline	28.5	43.3	16.4	33.9
Finetuned	52.9 (+24.4)	38.7 (-4.6)	30.5 (+14.1)	45.3 (+11.4)

Discussion. Table 16 shows that standard GRPO yields modest gains under contextual drag, improving over the baseline by $+1.9$ to $+5.0$ points across benchmarks, and substantially underperforming targeted SFT. This gap is consistent with GRPO's reward being defined solely on final-answer correctness: while it can nudge the policy toward better outcomes on average, it does not explicitly teach the model to *reset* its reasoning trajectory after detecting errors in the in-context draft.

In contrast, the supervised objective directly trains the post-verification trajectory to match the DIRECT reasoning trace, which appears more aligned with mitigating contextual drag. Overall, these results suggest that naïvely applying outcome-based RL is not a drop-in replacement for targeted supervision in this setting, and that more structured rewards (e.g., verifier-aware or trajectory-level signals) may be needed for RL-style training to reliably counteract erroneous context.