Same Evidence, Different Answers: Canonical-Context On-Policy Distillation for Multi-Turn Language Models

Anonymous ACL submission

Abstract

Large language models (LLMs) often solve a task when all instructions are given in a single prompt, but fail when the same information is revealed gradually across turns. When a clean FULL prompt and a RAW-SHARDED conversation contain the same complete user evidence, the model should still arrive at the same answer. We argue that a key reason for this gap is self-anchored drift: responses produced under partial information introduce unsupported assumptions, and those assumptions later distort the final answer. To reduce this effect, we propose Canonical-Context On-Policy Distillation (CCOPD), which uses the same base model in two roles. During training, the same base model is used in two roles: a frozen teacher conditioned on the clean FULL prompt and a trainable student that receives the same evidence incrementally through a multi-turn conversation; CCOPD aligns the student's behavior on its own trajectories with the teacher's canonical full-context behavior. Trained only on math problem conversations, CCOPD yields a 32% average relative improvement in RAW-SHARDED performance over the original base model across math and five zero-shot out-of-domain task families, while largely preserving full-context performance. Further analyses suggest that CCOPD strengthens grounding in user evidence and reduces sensitivity to contamination from earlier assistant turns.

1 Introduction

Users rarely specify a task in one fully formed prompt: they often reveal constraints over several turns. In multi-turn interactions where task-related user information is provided gradually, a fundamental reliability criterion requires the model to yield the same outcomes as it would when all such information is given in a single prompt after the conversation ends. We call this requirement *canonical-context consistency*. Recent multi-turn evaluations show that this consistency is fragile: models that

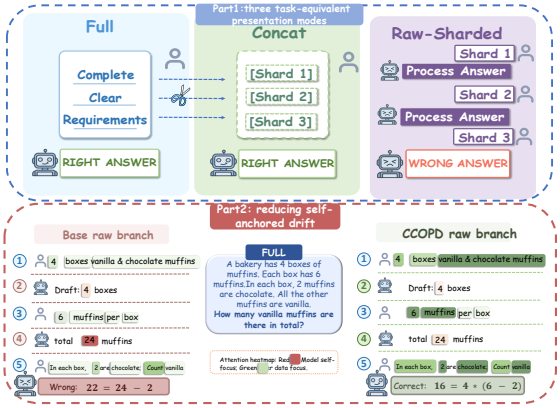


Figure 1: **Part 1**: Task-equivalent FULL, CONCAT, and RAW-SHARDED presentations. **Part 2**: Reduced self-anchored drift and improved canonical-context consistency.

succeed when a task is presented as a clean FULL prompt can still fail when the same information is disclosed incrementally through a RAW-SHARDED conversation (Laban et al., 2025; He et al., 2024; Li et al., 2025b). Part 1 of Figure 1 provides a concrete example of the three task-equivalent presentation modes.

The core difficulty is that a RAW-SHARDED history is not merely a longer prompt. It also contains the model's own earlier replies, produced before the task evidence is complete. Those replies can introduce tentative answers, unsupported assumptions, or partial reasoning into the context. By the final turn, the model may rely on this self-generated text instead of fully re-grounding its answer in the completed user-provided shards. We call this failure mode *self-anchored drift*.

A trivial solution is to repair the trajectory at inference time. Such methods can reflect on, revise, reset, or consolidate intermediate reasoning before the model commits to a final answer (Shinn et al., 2023; Madaan et al., 2023; Mohammad Khalid et al., 2025). Although they are often useful in deployment, they rely on an additional external

control loop and thus modify the inference process itself. Another line of work addresses the problem through clarification and abstention: when a request is ambiguous or the information needed for answering is still missing, the model must decide whether to ask a clarifying question, defer its response, or abstain (Zhang et al., 2025; Zhang and Choi, 2025; Testoni and Fern'andez, 2024; Wu et al., 2025; Zhang et al., 2024a; Cheng et al., 2024). However, recent evidence suggests that current off-the-shelf LLMs still struggle to reliably identify ambiguous requests and ask appropriate clarifying questions in practice, while still introducing contamination from the model's own prior responses (Zhang et al., 2024b; Li et al., 2025a; Luo et al., 2025).

We argue that **canonical-context consistency should be internalized as a core model capability**: once the final turn provides sufficient user evidence, the model's answer must be anchored to that evidence rather than deviating toward assumptions introduced by itself. To this end, we propose **Canonical-Context On-Policy Distillation** (CCOPD), a self-distillation objective that enforces canonical-context consistency under contaminated multi-turn histories. The same base model is adopted in two roles: a frozen teacher reads the clean canonical FULL task, while the trainable student receives the task only through the gradually revealed RAW-SHARDED interaction. Once all task-relevant evidence has been revealed through the interaction, the FULL-conditioned teacher provides token-level supervision on the student's own final-answer prefixes. This same-prefix, on-policy supervision re-anchors the RAW-SHARDED-conditioned student, encouraging its answer distribution to move away from self-generated contamination and toward the canonical distribution grounded in the completed user evidence. In this sense, CCOPD does not rely on a stronger external teacher. It targets cases where the same base model can solve the task under the task-equivalent clean FULL prompt, but may fail under the RAW-SHARDED history. CCOPD trains the model to preserve that FULL-context capability in the self-contaminated RAW-SHARDED setting.

Under the task-equivalent sharding setup introduced by Laban et al. (2025), CCOPD is trained only on sharded GSM8K-style math conversations. Strikingly, this math-only post-training signal transfers well beyond math. It improves RAW-SHARDED accuracy on math and yields zero-shot gains across five non-math settings. Overall, CCOPD yields a 32% average relative improvement over the unmodified base model on RAW-SHARDED performance, while largely preserving FULL and CONCAT performance.

Our contributions are:

- We formalize canonical-context consistency for task-equivalent histories and identify self-anchored drift as a concrete source of its failure. We introduce CCOPD, an on-policy same-prefix distillation objective that aligns RAW-SHARDED final-answer behavior with a frozen same-backbone FULL teacher, without adding a stronger external teacher or an inference-time repair loop.
- We show that a math-only CCOPD signal transfers beyond math: the same adapter improves RAW-SHARDED performance across all six task families, including five non-math zero-shot settings, while largely preserving FULL and CONCAT performance. These gains exceed the data-matched post-training baselines in our evaluation.
- We support the proposed mechanism with a suite of ablations and diagnostics. These analyses show that the gains are tied to canonical FULL-view alignment and reverse-KL same-prefix supervision; when the model's own FULL behavior is already strong, increasing teacher scale brings limited additional benefit. Matched pollution tests and evidence-focus probes further show stronger grounding in completed user evidence and reduced sensitivity to assistant-side anchors.

2 Related Work

Inference-time repair and control loops. Inference-time repair and self-correction methods address model errors by adding an explicit test-time control process around generation (Pan et al., 2024; Kamoi et al., 2024). Representative approaches revise generations through verbal reflection or self-feedback (Shinn et al., 2023; Madaan et al., 2023; Ferraz et al., 2024), check candidate answers with self-verification, external critics, or learned verifiers (Weng et al., 2023; Gou et al., 2024; Zhang et al., 2024c; Gao et al., 2023), or reset the context before regenerating (Mohammad Khalid et al., 2025). These methods can be effective when extra inference passes and

reliable feedback are available, but they optimize an augmented test-time system. Our setting is complementary: the model must answer from the completed transcript itself, where earlier partial-information replies may remain as misleading context. CCOPD targets this robustness through training, rather than by adding a repair or reset loop at test time.

Clarification, abstention, and online interaction control. Another line of work studies how models should behave when a request is ambiguous or the information needed to answer is incomplete. Selective-prediction and ambiguity-aware generation methods study when models should withhold unreliable answers (Cole et al., 2023; Kim et al., 2024). Clarification work evaluates when to ask a question and what question to ask (Zhang et al., 2024b; Zhang and Choi, 2025). This literature asks whether to answer, ask, defer, or abstain under uncertainty. Our setting assumes that the missing evidence has already arrived, but the realized transcript may still contain assistant-side commitments made before it was complete. CCOPD targets how the model conditions on a completed but self-contaminated transcript.

On-policy distillation (OPD). On-policy distillation reduces the trajectory mismatch in autoregressive distillation by supervising the student on states reached by its own policy, rather than only on fixed teacher demonstrations (Gu et al., 2024; Agarwal et al., 2024). Many recent OPD variants strengthen the supervision signal by giving the teacher access to additional task information (Zhao et al., 2026a; Penaloza et al., 2026; Liu et al., 2026; Ye et al., 2026; Zhang et al., 2026), or by coupling distillation with auxiliary inference, feedback, or optimization mechanisms (Zhao et al., 2026b; Yang et al., 2026; Zhu et al., 2026). CCOPD uses OPD for a different purpose. The teacher and student are given the same underlying task evidence and share the same backbone, but they condition on different presentations of that evidence: a clean FULL for the teacher and a realized RAW-SHARDED transcript for the student. Thus, the teacher is presentation-privileged rather than information-privileged. The goal is not to import extra knowledge from a stronger or better-informed supervisor, but to distill invariance across task-equivalent presentations.

3 Problem Statement

3.1 Problem Formulation

Task-equivalent sharded presentations. For a task instance q, let $c(q)$ denote the canonical FULL prompt. Let $h(q)$ denote a sharded presentation of the same task, where task information is released across user turns and interleaved with assistant replies before the final-answer request. We call $h(q)$ *task-equivalent* to $c(q)$ when the sharded presentation reveals the same task-relevant user evidence as the canonical prompt.

Canonical Context Consistency. For task-equivalent presentations, a reliable model should preserve its final-answer distribution under the change of presentation:

$$\pi(y \mid h(q)) \approx \pi(y \mid c(q)). \tag{1}$$

We call this requirement *canonical-context consistency*. The canonical prompt is not assumed to be an oracle; it is a controlled presentation in which the same user evidence is available.

To inspect where this invariance breaks down, we compare next-token predictions under the two task-equivalent presentations while holding the answer prefix fixed. For a shared answer prefix s, define

$$\Psi_\pi(q, s) = D_{\mathrm{KL}}(\pi(\cdot \mid h(q), s) \parallel \pi(\cdot \mid c(q), s)). \tag{2}$$

The same model appears on both sides; only the preceding presentation changes. Hence Ψ_π is a local diagnostic of presentation sensitivity. Large values mark prefixes where the sharded path induces continuation preferences that differ from those induced by the clean presentation.

Self-anchored drift. The discrepancy in Eq. (2) locates a shift but does not identify which part of the sharded presentation is responsible. In RAW-SHARDED histories, the distinctive source of concern is earlier assistant text generated before the task evidence is complete. We write a final-turn RAW-SHARDED history as

$$h_{\text{RAW-SHARDED}}(q) = (u_1, a_1, \ldots, u_{K-1}, a_{K-1}, u_K), \tag{3}$$

where u_i are user shards and a_i are non-final assistant replies. Each a_i is generated before all task evidence is available, and may therefore contain unsupported guesses, provisional answers, or task-specific commitments. After u_K supplies the missing evidence, the user evidence in the history is

complete and task-equivalent to $c(q)$. However, the earlier a_i remain in the conditioning context. When these self-generated commitments pull the final-answer behavior away from the completed user evidence, canonical-context consistency fails. We call this failure mode *self-anchored drift*.

3.2 Probing self-anchored drift

We probe the unmodified base model to test whether non-final assistant replies still affect the final predictive state after all user evidence is present. We refer to these non-final replies as process replies.

SAAR and span edit. For completed RAW-SHARDED histories, we annotate completed user-evidence spans G_{usr} and assistant commitment spans G_{self}. The *Self-Anchor Attention Ratio* (SAAR) summarizes final-answer attention over these spans:

$$\text{SAAR}_\ell = \log \frac{\bar{A}_\ell(G_{\text{usr}}) + \epsilon}{\bar{A}_\ell(G_{\text{self}}) + \epsilon}. \tag{4}$$

Here $\bar{A}_\ell(G)$ averages layer-ℓ attention from answer tokens to group G; higher SAAR indicates denser attention to completed user evidence relative to earlier assistant commitments. Using the same spans, we also mask commitment spans and measure the change in the gold-vs-process-anchor margin. Figure 2a shows that this edit selectively helps states anchored to the model's process-stage wrong answer.

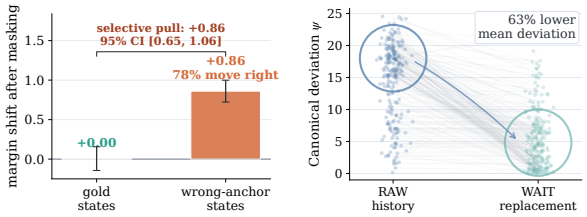
Neutral-placeholder contrast. We replace each process reply with a neutral placeholder while keeping the user shards and turn structure fixed. For prefix i, let C_i^{raw}, C_i^{neu}, and C_i^{full} be the realized, neutralized, and canonical FULL contexts. With $p_{i,r}^M = p_M(\cdot \mid C_i^r)$ for $r \in \{\text{raw}, \text{neu}\}$ and $q_i^{\text{full}} = \pi_0(\cdot \mid C_i^{\text{full}})$, define

$$\Delta_i^M = D_{\text{KL}}(p_{i,\text{raw}}^M \,\|\, q_i^{\text{full}}) - D_{\text{KL}}(p_{i,\text{neu}}^M \,\|\, q_i^{\text{full}}). \tag{5}$$

Positive Δ_i^M means that process replies increase canonical deviation; Figure 2b shows that neutralization reduces it.

4 Canonical-Context On-Policy Distillation

Section 3 defines a local presentation gap at a shared answer prefix. CCOPD turns this gap into an on-policy distillation signal: a trainable student



(a) Span edit (b) Neutral-placeholder replacement

Figure 2: Qwen3-8b-base model probes for self-anchored drift. (a) Masking process-reply commitment spans selectively improves gold-vs-anchor margins for wrong-anchor states. (b) Replacing process replies with neutral placeholders lowers predictive-state deviation from the canonical FULL reference.

answers from the realized RAW-SHARDED history, while a frozen copy of the same base model scores the student's own answer prefixes under the canonical FULL prompt. Training minimizes a final-answer-masked reverse KL between these two next-token distributions.

4.1 Retained Pairs and Model Roles

For each task q, we form a final-turn pair (c, h), where $c = c(q)$ and h is the retained sharded history just before the final answer. In the main setting, $h = h_{\text{RAW-SHARDED}}(q)$: all task-relevant user shards have been revealed, and the non-final assistant replies from the realized interaction remain in the context. We retain only histories that end with the final user turn after the user evidence is complete.

The two contexts are used on separate paths. The student is trainable and conditions only on h; the canonical prompt c is never appended to the student input. The teacher is a frozen copy of the same base model and conditions only on c. Thus, CCOPD compares two presentations of the same user evidence while updating only the history-conditioned student. Appendix A.2 details the structural checks and leakage controls.

4.2 On-Policy Canonical Relabeling

Given a retained pair (c, h), the current student produces a final-answer rollout

$$\hat{y}_{1:T} \sim \pi_\theta(\cdot \mid h), \tag{6}$$

and $T_{\text{ans}}(\hat{y})$ denotes the token positions belonging to that final assistant continuation. These prefixes are the on-policy states for distillation: they are

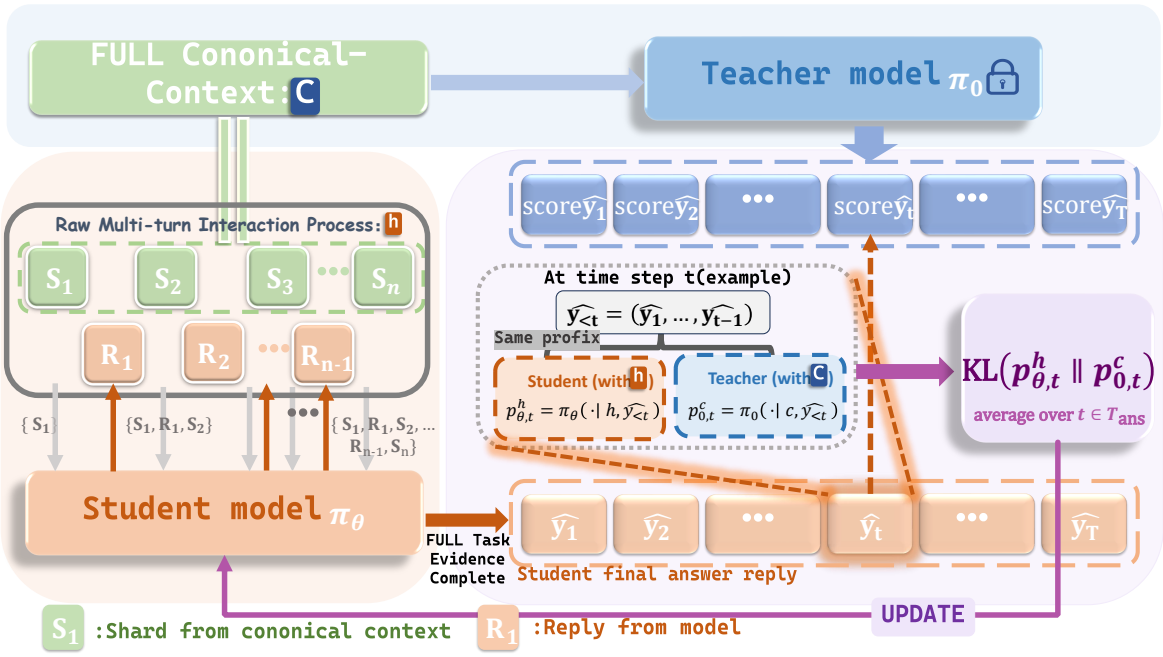


Figure 3: Overview method of CCOPD. CCOPD aligns a RAW-SHARDED-conditioned student with a frozen FULL-conditioned teacher using answer-masked same-prefix reverse KL.

reached after the student conditions on the realized transcript, including any earlier assistant-side commitments contained in h.

For each $t \in T_{\text{ans}}(\hat{y})$, the student and teacher score the same prefix $\hat{y}_{<t}$ under different contexts:

$$p^h_{\theta,t} = \pi_\theta(\cdot \mid h, \hat{y}_{<t}), \quad p^c_{0,t} = \pi_0(\cdot \mid c, \hat{y}_{<t}). \quad (7)$$

The teacher is not asked to decode a clean completion. It only provides next-token probabilities on the prefix that the student has already visited. Thus canonical relabeling changes the conditioning context of the scorer, not the answer prefix being scored.

As in on-policy distillation, the rollout operation in Equation (6) selects the states but is treated as stop-gradient. Gradients flow through the next-token student distribution evaluated at the selected prefixes.

4.3 Answer-Masked Reverse-KL Objective

Substituting Equation (7) into the canonical-deviation potential gives the average deviation along the student rollout:

$$\bar{\Psi}_\theta(q, \hat{y}) = \frac{1}{|T_{\text{ans}}(\hat{y})|} \sum_{t \in T_{\text{ans}}(\hat{y})} D_{\mathrm{KL}}\left(p^h_{\theta,t} \,\middle\|\, p^c_{0,t}\right). \quad (8)$$

The implemented per-rollout loss is this quantity,

$$\ell_{\text{ccopd}}(q, \hat{y}) = \bar{\Psi}_\theta(q, \hat{y}), \quad (9)$$

and the ideal objective is

$$\mathcal{L}_{\text{ccopd}}(\theta) = \mathbb{E}_{(c,h)\sim\mathcal{D}_{\text{pair}}} \, \mathbb{E}_{\hat{y}\sim\pi_\theta(\cdot|h)} \left[\ell_{\text{ccopd}}(q, \hat{y})\right]. \quad (10)$$

The finite training loss estimates Equation (10) by averaging Equation (9) over minibatches. Normalizing by $|T_{\text{ans}}(\hat{y})|$ prevents long answers from dominating the objective.

The final-answer mask is part of the objective, not an implementation detail. Intermediate assistant turns in a RAW-SHARDED conversation are produced before all user evidence is available; forcing those turns to match a FULL-conditioned teacher would leak future evidence and penalize reasonable partial-information behavior. CCOPD therefore optimizes canonical alignment only after the final user turn, where the history and the canonical prompt are task-equivalent.

Sequence-level view. Let P^h_θ be the distribution over complete final answers induced by the student under h, and let P^c_0 be the analogous distribution induced by the frozen base model under c. Under an ideal stochastic formulation, the autoregressive chain rule decomposes $D_{\mathrm{KL}}(P^h_\theta \| P^c_0)$ into the expected sum of the same-prefix token KL terms along student-sampled trajectories. Equation (10) can therefore be read as a practical, answer-masked estimator of answer-level alignment to the canonical full-context behavior. Appendix A states the

exact bridge and its assumptions.

Figure 3 shows training step. The important asymmetry is that the student path and teacher path receive different contexts but score the same prefix. This yields a signal that is both cross-context and on-policy.

5 Experiments and Analysis

We evaluate whether CCOPD improves canonical-context consistency after the user evidence is complete. The target setting is RAW-SHARDED; FULL measures clean one-shot capability, and CONCAT checks whether the released shards preserve the task information.

5.1 Experimental Setup

Models. Our primary experiments use Qwen3-8B with a LoRA adapter trained by the reverse-KL CCOPD objective in Equation (10) (Team et al., 2025; Hu et al., 2022). For the main Qwen3-8B run, we optimize the LoRA adapter with AdamW using a learning rate of 3×10^{-5}; full training details are reported in Table 10. To assess scale and family robustness, we additionally train adapters for Qwen3-4B and Llama3.1-8B (Grattafiori et al., 2024) in the scaling analysis.

Training data and pair construction. Training is restricted to math. We construct sharded conversations from GSM8K and GSM8K-Aug (Cobbe et al., 2021; Deng et al., 2024). For each base model, we generate on-policy RAW-SHARDED interactions from the same shard specification.The primary Qwen3-8B adapter is trained on 6k retained pairs after screening.

Evaluation protocol. We follow the released sharded-instruction protocol from Lost in Conversation (Laban et al., 2025). The evaluation suite covers six task families: math word problems from GSM8K-style examples (Cobbe et al., 2021), code generation from HumanEval and LiveCodeBench-derived tests (Chen et al., 2021; Jain et al., 2024), function calling from BFCL-style tasks (Patil et al., 2025), text-to-SQL from Spider (Yu et al., 2018), table-to-text generation from ToTTo (Parikh et al., 2020), and long-context summarization from SummHay (Laban et al., 2024). The non-math tasks are zero-shot transfer settings for the math-trained adapter. We report the mean over 10 independent end-to-end sharded runs with newly sampled on-policy conversations. details are given in Appendix B.

Compared systems. **Base** is the unmodified backbone and defines deltas. For data-matched training comparisons, **SFT** uses the same retained final-turn histories h and the same final-answer mask as CCOPD, but replaces distributional alignment with supervised final-answer targets; **GRPO** uses the same math source and final-answer reward/evaluation protocol. **OPSD** is a related on-policy distillation comparison with privileged supervision. **Forward-CCOPD** and **CONCAT-teacher** are token-level controls that keep the retained histories, final-answer positions, and student prefixes fixed, while changing only the KL direction or teacher conditioning context. **Teacher-source** rows replace the same-backbone FULL teacher with smaller or larger teachers.

5.2 Main Results

Table 1 summarizes the main results; Table 4 gives per-domain details.

RAW improves without clean-task drift. On Qwen3-8B, CCOPD raises math RAW-SHARDED accuracy from 66.0 to 82.5 (+16.5 points), while math FULL remains unchanged at 90.3. CCOPD also outperforms SFT and GRPO on math RAW-SHARDED, suggesting that the observed RAW-SHARDED gains are not recovered by standard post-training objectives alone.

Math-only training transfers out of domain. The same math-trained adapter improves the non-math RAW-SHARDED average from 36.8 to 49.7 (+12.9 points). Structured OOD RAW-SHARDED rises from 48.7 to 67.9 (+19.2), and generation OOD RAW-SHARDED rises from 19.4 to 22.9 (+3.5). Because training uses only math conversations, this pattern is more consistent with transferable history grounding than with a math-format heuristic. Qwen3-4B and Llama3.1-8B show the same aggregate RAW-SHARDED direction.

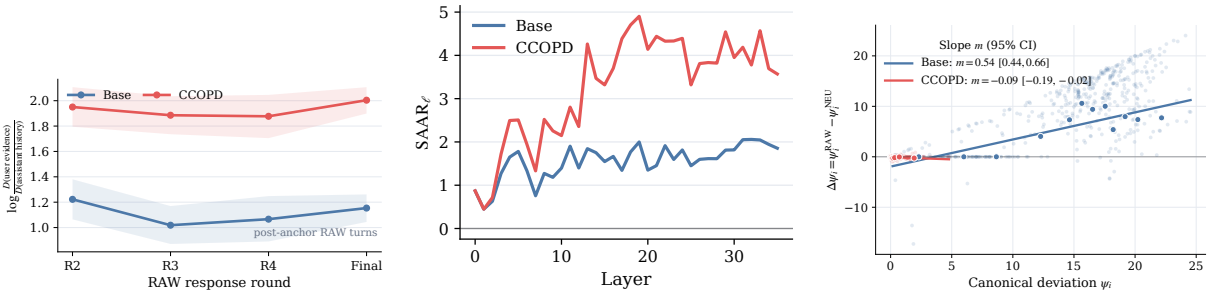
We further train CCOPD on non-math HotpotQA(Yang et al., 2018) sharded histories and still observe an improvement on math RAW-SHARDED , suggesting that the signal is not tied to math-format supervision alone; details are in Appendix E.

5.3 Ablations and Mechanism Diagnostics

Teacher source. Teacher-source ablations in Table 1 suggest that scale is not the main bottleneck. A smaller 4B FULL teacher still improves the weighted RAW-SHARDED aggregate over the base model, while a larger 14B teacher improves

Model / variant	Math F/C/R	Structured OOD F/C/R	Generation OOD F/C/R	All F/C	OOD-R	All-R
Model-size and model-family transfer						
Qwen3-14B Base	96.1 / 92.9 / 71.8	92.1 / 83.2 / 52.6	34.1 / 31.9 / 20.3	70.3	39.5	44.8
Qwen3-4B Base	89.3 / 84.5 / 55.3	80.1 / 74.0 / 41.0	22.4 / 22.6 / 11.5	60.2	29.1	33.4
Qwen3-4B CCOPD	89.3 / 83.5 / 56.9 (+1.6)	83.7 / 76.0 / 63.8	21.0 / 18.4 / 17.2	60.6 (+0.3)	**44.9 (+15.9)**	**46.9 (+13.5)**
Llama3.1-8B Base	71.8 / 70.9 / 46.6	67.6 / 63.7 / 37.5	13.0 / 16.3 / 13.1	49.3	27.6	30.8
Llama3.1-8B CCOPD	74.8 / 73.8 / 53.4 (+6.8)	67.3 / 64.1 / 52.7	14.8 / 13.8 / 13.4	49.7 (+0.4)	**36.8 (+9.1)**	**39.5 (+8.8)**
Qwen3-8B: objectives and teacher sources						
Qwen3-8B Forward-CCOPD	93.2 / 91.3 / 79.0 (+13.0)	84.6 / 76.0 / 65.4	27.0 / 23.3 / 20.4	63.6 (-0.8)	47.2 (+10.4)	52.4 (+10.8)
Qwen3-8B 4B-teacher CCOPD	90.3 / 88.4 / 68.0 (+1.9)	85.6 / 78.8 / 67.6	26.4 / 23.2 / 21.1	64.0 (-0.4)	48.8 (+12.0)	51.9 (+10.3)
Qwen3-8B 14B-teacher CCOPD	92.2 / 86.4 / 69.9 (+3.9)	84.6 / 78.2 / 66.3	27.8 / 23.9 / 20.6	63.9 (-0.5)	47.8 (+11.0)	51.4 (+9.8)
Qwen3-8B: complete baselines						
Qwen3-8B Base	90.3 / 87.4 / 66.0	85.3 / 78.8 / 48.7	28.4 / 24.7 / 19.4	64.4	36.8	41.6
Qwen3-8B SFT	93.2 / 91.3 / 66.0 (+0.0)	86.9 / 76.9 / 64.4	27.4 / 22.4 / 19.0	64.3 (-0.1)	46.0 (+9.2)	49.3 (+7.7)
Qwen3-8B OPSD	86.0 / 85.0 / 56.3 (-9.7)	87.8 / 77.9 / 54.8	27.7 / 23.8 / 17.7	64.0 (-0.4)	39.8 (+3.0)	42.5 (+0.9)
Qwen3-8B GRPO	91.3 / 90.3 / 74.8 (+8.8)	86.9 / 84.0 / 45.8	28.4 / 24.9 / 19.2	66.4 (+2.0)	35.0 (-1.8)	41.6 (-0.1)
Qwen3-8B CCOPD	90.3 / 88.4 / **82.5 (+16.5)**	86.2 / 77.9 / **67.9**	27.9 / 23.3 / **22.9**	64.2 (-0.2)	**49.7 (+12.9)**	**55.1 (+13.5)**

Table 1: Master results with weighted aggregate scores. F/C/R denotes FULL/CONCAT/RAW-SHARDED. Structured OOD averages code, function calling, and text-to-SQL; Generation OOD averages table-to-text and summarization. All F/C is a single clean-presentation score obtained by averaging the weighted FULL and weighted CONCAT scores across all six task families. OOD-R averages non-math RAW-SHARDED scores; All-R averages all six task families. Parenthesized values indicate changes against the matching base row under the same weighted scope; in Math F/C/R, only the R component carries the parenthesized RAW delta.



(a) Turn-level evidence focus (b) Final-state SAAR (c) Neutral replacement

Figure 4: Mechanism diagnostics for self-anchored drift. (a) CCOPD maintains stronger user-evidence focus across RAW-SHARDED response rounds after process replies enter the context. (b–c) At matched final-answer prefixes, CCOPD increases SAAR and flattens assistant-associated canonical deviation, reducing the neutral-replacement slope from 0.537 to -0.087.

Condition	Base	CCOPD
Clean FULL problem	90.3	90.3
+ Assistant-side wrong solution	33.0	89.3
+ User-side wrong answer hint	63.1	88.3

Table 2: Full-context pollution stress test on math.

over the base but remains below the same-backbone teacher. This pattern supports the intended role of the teacher in CCOPD: it need not supply new task knowledge, but should provide a compatible canonical presentation of the same evidence. Cross-model mismatch may partly limit external teachers, especially at larger scale.

KL direction. Forward-CCOPD keeps the student rollouts, FULL teacher, final-answer mask, and same-prefix scoring fixed, but replaces reverse KL with forward KL. This control tests both whether fine-grained same-prefix canonical supervision helps over the base model and whether the KL direction matters beyond that supervision. Forward-CCOPD improves substantially over the base, but remains below reverse-KL CCOPD. We attribute the gap to the objectives: forward KL encourages the RAW-SHARDED student to cover the FULL teacher's likely continuations, whereas reverse KL penalizes off-canonical probability mass assigned by the deployed RAW-SHARDED policy. Because self-anchored drift appears in the policy used at inference time, reverse KL better matches the intended intervention.

History-stress diagnostics. We test grounding under matched polluted histories. Starting from full math problems, we insert the same per-example wrong numeric anchor either as an assistant-side prior solution or as a user-side answer hint; within each condition, both models receive the same evaluated history. Table 2 shows that the two models match on the clean FULL condition, but the base model collapses under pollution ($90.3 \rightarrow 33.0/63.1$), whereas CCOPD remains near clean performance ($90.3 \rightarrow 89.3/88.3$). This suggests reduced sensitivity to misleading anchors when complete user evidence is available.

Evidence focus across process and final-answer states. We separate trajectory-level evidence focus from final-state anchoring in Figure 4. At the RAW-SHARDED trajectory level, Figure 4a tracks attention allocation across response rounds: after process replies enter the history, CCOPD maintains higher relative attention to task-relevant user evidence than to process history. At completed final-answer states, Figure 4b–c apply the same-prefix diagnostics from Section 3: SAAR compares completed user-evidence spans with process-reply commitment spans, and neutral replacement measures assistant-associated canonical deviation under fixed user shards. CCOPD raises layer SAAR and flattens the neutral-replacement lower-envelope slope from 0.537 to -0.087. Together, these probes are consistent with reduced self-anchored drift: the trained model attends more to completed user evidence and shows less assistant-associated deviation at the final answer.

Lightweight test-time modes. We additionally evaluate two lightweight test-time modes: reset-then-answer (RTA) and defer-until-complete (DUC). As shown in Table 3, direct CCOPD is the strongest setting on math RAW-SHARDED. RTA helps the base model, whereas DUC has little effect. Applying the same modes to CCOPD slightly lowers accuracy, although both variants remain above the corresponding base+mode rows. This pattern is consistent with the possibility that CCOPD has already internalized part of the task-state management that these prompts impose explicitly. Implementation details are provided in Appendix H.

Model	Default	+RTA	+DUC
Base	66.0	72.8	67.0
CCOPD	**82.5**	79.6	77.7

Table 3: Accuracy on math RAW-SHARDED under two lightweight test-time modes.

6 Conclusion

This paper reframes RAW-SHARDED multi-turn failure as self-anchored canonical-context drift. The final history may contain all user evidence, but earlier assistant assumptions can still pull the model away from the answer distribution it would use under a clean FULL prompt. CCOPD addresses this failure by distilling RAW-SHARDED final-answer states toward a frozen FULL teacher on the same generated prefix. The method adds no inference-time component, improves on-policy sharded accuracy after math-only training, and transfers zero-shot to five non-math task types. Attention redistribution and WAIT replacement diagnostics suggest that the trained model becomes less sensitive to self-generated assumptions and more aligned with user evidence at the final answer.

Limitations

Our experiments are still moderate in scale. The primary trained model is an 8B open-weight backbone with a LoRA adapter, trained on thousands rather than millions of retained canonical/history pairs. While this scale is sufficient to test the proposed mechanism, it does not establish how CCOPD behaves for substantially larger models, closed-source systems, or broader post-training corpora. Our RAW-SHARDED setting is also a controlled approximation of multi-turn interaction. It is constructed by splitting complete benchmark tasks into shards, which enables a clean task-equivalent comparison against FULL and CONCAT prompts. However, real conversations may contain evolving intents, clarification requests, irrelevant side information, user corrections, and failures that cannot be reduced to deterministic shards of a complete prompt. Future work should collect organic multi-turn failures, construct human-verified canonical contexts that make those tasks answerable, and use such paired data to study self-anchored drift in more natural settings.Finally, CCOPD depends on a useful FULL-conditioned reference and only optimizes the final-answer state after all user evi-

dence is available. It can improve consistency with the model's clean full-context behavior, but it cannot correct errors made by the FULL teacher itself, nor does it directly train earlier-turn behaviors such as clarification, abstention, or avoiding premature commitments under incomplete information.

References

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. 2024. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations*, volume 2024, pages 21246–21263.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 39 others. 2021. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

Qinyuan Cheng, Tianxiang Sun, Xiangyang Liu, Wenwei Zhang, Zhangyue Yin, Shimin Li, Linyang Li, Zhengfu He, Kai Chen, and Xipeng Qiu. 2024. Can AI assistants know what they don't know? In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 8184–8202. PMLR.

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

Jeremy Cole, Michael Zhang, Daniel Gillick, Julian Eisenschlos, Bhuwan Dhingra, and Jacob Eisenstein. 2023. Selectively answering ambiguous questions. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 530–543, Singapore. Association for Computational Linguistics.

Yuntian Deng, Yejin Choi, and Stuart Shieber. 2024. From explicit cot to implicit cot: Learning to internalize cot step by step. *arXiv preprint arXiv:2405.14838*.

Thomas Palmeira Ferraz, Kartik Mehta, Yu-Hsiang Lin, Haw-Shiuan Chang, Shereen Oraby, Sijia Liu, Vivek Subramanian, Tagyoung Chung, Mohit Bansal, and Nanyun Peng. 2024. LLM self-correction with DeCRIM: Decompose, critique, and refine for enhanced following of instructions with multiple constraints. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 7773–7812, Miami, Florida, USA. Association for Computational Linguistics.

Luyu Gao, Zhuyun Dai, Panupong Pasupat, Anthony Chen, Arun Tejasvi Chaganty, Yicheng Fan, Vincent Y. Zhao, Ni Lao, Hongrae Lee, Da-Cheng Juan, and Kelvin Guu. 2023. Rarr: Researching and revising what language models say, using language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16477–16508, Toronto, Canada. Association for Computational Linguistics.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. 2024. Critic: Large language models can self-correct with tool-interactive critiquing. In *The Twelfth International Conference on Learning Representations*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. Technical report.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2024. MiniLLM: Knowledge distillation of large language models. In *The Twelfth International Conference on Learning Representations*.

Yun He, Di Jin, Chaoqi Wang, Chloe Bi, Karishma Mandyam, Hejia Zhang, Chen Zhu, Ning Li, Tengyu Xu, Hongjiang Lv, Shruti Bhosale, Chenguang Zhu, Karthik Abinav Sankararaman, Eryk Helenowski, Melanie Kambadur, Aditya Tayade, Hao Ma, Han Fang, and Sinong Wang. 2024. Multi-IF: Benchmarking LLMs on multi-turn and multilingual instructions following. *Preprint*, arXiv:2410.15553.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. 2024. LiveCodeBench: Holistic and contamination free evaluation of large language models for code. *Preprint*, arXiv:2403.07974.

Ryo Kamoi, Yusen Zhang, Nan Zhang, Jiawei Han, and Rui Zhang. 2024. When can LLMs actually correct their own mistakes? a critical survey of self-correction of LLMs. *Transactions of the Association for Computational Linguistics*, 12:1417–1440.

Hyuhng Joon Kim, Youna Kim, Cheonbok Park, Junyeob Kim, Choonghyun Park, Kang Min Yoo, Sanggoo Lee, and Taeuk Kim. 2024. Aligning language models to explicitly handle ambiguity. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1989–2007, Miami, Florida, USA. Association for Computational Linguistics.

Philippe Laban, Alexander Fabbri, Caiming Xiong, and Chien-Sheng Wu. 2024. Summary of a haystack: A challenge to long-context LLMs and RAG systems. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 9885–9903, Miami, Florida, USA. Association for Computational Linguistics.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. 2025. LLMs get lost in multi-turn conversation. *CoRR*, abs/2505.06120.

Belinda Z. Li, Been Kim, and Zi Wang. 2025a. Questbench: Can LLMs ask the right question to acquire information in reasoning tasks? *Preprint*, arXiv:2503.22674.

Jinnan Li, Jinzhe Li, Yue Wang, Yi Chang, and Yuan Wu. 2025b. StructFlowBench: A structured flow benchmark for multi-turn instruction following. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 9322–9341, Vienna, Austria. Association for Computational Linguistics.

Yihong Liu, Raoyuan Zhao, Michael A Hedderich, and Hinrich Schütze. 2026. Crosslingual on-policy self-distillation for multilingual reasoning. *arXiv preprint arXiv:2605.09548*.

Sichun Luo, Yi Huang, Mukai Li, Shichang Meng, Fengyuan Liu, Zefa Hu, Junlan Feng, and Qi Liu. 2025. Clarifymt-bench: Benchmarking and improving multi-turn clarification for conversational large language models. *Preprint*, arXiv:2512.21120.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, volume 36.

Haziq Mohammad Khalid, Athikash Jeyaganthan, Timothy Do, Yicheng Fu, Vasu Sharma, Sean O'Brien, and Kevin Zhu. 2025. ERGO: Entropy-guided resetting for generation optimization in multi-turn language models. In *Proceedings of the 2nd Workshop on Uncertainty-Aware NLP (UncertaiNLP 2025)*, pages 273–286, Suzhou, China. Association for Computational Linguistics.

Liangming Pan, Michael Saxon, Wenda Xu, Deepak Nathani, Xinyi Wang, and William Yang Wang. 2024. Automatically correcting large language models: Surveying the landscape of diverse automated correction strategies. *Transactions of the Association for Computational Linguistics*, 12:484–506.

Ankur Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. 2020. ToTTo: A controlled table-to-text generation dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1173–1186, Online. Association for Computational Linguistics.

Shishir G Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. 2025. The berkeley function calling leaderboard (BFCL): From tool use to agentic evaluation of large language models. In *Forty-second International Conference on Machine Learning*.

Emiliano Penaloza, Dheeraj Vattikonda, Nicolas Gontier, Alexandre Lacoste, Laurent Charlin, and Massimo Caccia. 2026. Privileged information distillation for language models. *arXiv preprint arXiv:2602.04942*.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36.

Qwen Team, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *arXiv (Cornell University)*.

Alberto Testoni and Raquel Fern'andez. 2024. Asking the right question at the right time: Human and model uncertainty guidance to ask clarification questions. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 258–275. Association for Computational Linguistics.

Yixuan Weng, Minjun Zhu, Fei Xia, Bin Li, Shizhu He, Shengping Liu, Bin Sun, Kang Liu, and Jun Zhao. 2023. Large language models are better reasoners with self-verification. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 2550–2575, Singapore. Association for Computational Linguistics.

Shirley Wu, Michel Galley, Baolin Peng, Hao Cheng, Gavin Li, Yao Dou, Weixin Cai, James Zou, Jure Leskovec, and Jianfeng Gao. 2025. CollabLLM: From passive responders to active collaborators. In *Proceedings of the 42nd International Conference on Machine Learning*.

Wenkai Yang, Weijie Liu, Ruobing Xie, Kai Yang, Saiyong Yang, and Yankai Lin. 2026. Learning beyond teacher: Generalized on-policy distillation with reward extrapolation. *arXiv preprint arXiv:2602.12125*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Tianzhu Ye, Li Dong, Xun Wu, Shaohan Huang, and Furu Wei. 2026. On-policy context distillation for language models. *Preprint*, arXiv:2602.12275.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir R. Radev. 2018. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921, Brussels, Belgium. Association for Computational Linguistics.

Hanning Zhang, Shizhe Diao, Yong Lin, Yi Fung, Qing Lian, Xingyao Wang, Yangyi Chen, Heng Ji, and Tong Zhang. 2024a. R-tuning: Instructing large language models to say 'i don't know'. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7113–7139, Mexico City, Mexico. Association for Computational Linguistics.

Michael JQ Zhang and Eunsol Choi. 2025. Clarify when necessary: Resolving ambiguity through interaction with LMs. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 5541–5558, Albuquerque, New Mexico. Association for Computational Linguistics.

Michael JQ Zhang, W. Bradley Knox, and Eunsol Choi. 2025. Modeling future conversation turns to teach LLMs to ask clarifying questions. In *International Conference on Learning Representations*.

Tong Zhang, Peixin Qin, Yang Deng, Chen Huang, Wenqiang Lei, Junhong Liu, Dingnan Jin, Hongru Liang, and Tat-Seng Chua. 2024b. CLAMBER: A benchmark of identifying and clarifying ambiguous information needs in large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10746–10766, Bangkok, Thailand. Association for Computational Linguistics.

Xinsen Zhang, Zhenkai Ding, Tianjun Pan, Run Yang, Chun Kang, Xue Xiong, and Jingnan Gu. 2026. Opsdl: On-policy self-distillation for long-context language models. *arXiv preprint arXiv:2604.17535*.

Yunxiang Zhang, Muhammad Khalifa, Lajanugen Logeswaran, Jaekyeom Kim, Moontae Lee, Honglak Lee, and Lu Wang. 2024c. Small language models need strong verifiers to self-correct reasoning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 15637–15653, Bangkok, Thailand. Association for Computational Linguistics.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. 2026a. Self-distilled reasoner: On-policy self-distillation for large language models. *Preprint*, arXiv:2601.18734.

Zhengyang Zhao, Lu Ma, and Wentao Zhang. 2026b. Training with harnesses: On-policy harness self-distillation for complex reasoning. *arXiv preprint arXiv:2605.08741*.

Siqi Zhu, Xuyan Ye, Hongyu Lu, Weiye Shi, and Ge Liu. 2026. The many faces of on-policy distillation: Pitfalls, mechanisms, and fixes. *arXiv preprint arXiv:2605.11182*.

A Proof Details and Implementation Notes for CCOPD

A.1 Answer-Event Control

Fix a task instance q. Write $h = h(q)$ for the completed multi-turn history and $c = c(q)$ for the canonical FULL prompt. Let P_θ^h be the distribution over complete final-answer strings induced by $\pi_\theta(\cdot \mid h)$, and let P_0^c be the corresponding distribution induced by $\pi_0(\cdot \mid c)$. We include EOS in the answer vocabulary and map any sequence that reaches the generation budget before EOS to a distinguished truncation symbol. Under this convention, both distributions are defined over the same finite terminal-answer space $\mathcal{A}$. We also assume $P_\theta^h \ll P_0^c$ on $\mathcal{A}$, so the KL terms below are well defined.

Lemma 1 (On-policy chain rule)**.** *For $y \sim P_\theta^h$ with terminal length $\tau(y)$,*

$$D_{\mathrm{KL}}(P_\theta^h \| P_0^c) = \mathbb{E}_{y \sim P_\theta^h}\left[\sum_{t=1}^{\tau(y)} d_t(y)\right], \tag{11}$$

where

$$d_t(y) = D_{\mathrm{KL}}(\pi_\theta(\cdot \mid h, y_{<t}) \| \pi_0(\cdot \mid c, y_{<t}))\,. \tag{12}$$

Proof. For any terminal answer string $y = (y_1, \ldots, y_{\tau(y)}) \in \mathcal{A}$, the two autoregressive distributions factor as

$$P_\theta^h(y) = \prod_{t=1}^{\tau(y)} \pi_\theta(y_t \mid h, y_{<t}), \tag{13}$$

$$P_0^c(y) = \prod_{t=1}^{\tau(y)} \pi_0(y_t \mid c, y_{<t}). \tag{14}$$

Substituting these factorizations into sequence-

Model	Variant	Code F/C/R	Function F/C/R	Text-to-SQL F/C/R	Table-to-text F/C/R	Summary F/C/R
Model-size and model-family transfer						
Qwen3-14B	Base	89.0 / 72.0 / 47.0	98.1 / 99.1 / 55.2	89.0 / 78.0 / 55.1	45.1 / 43.2 / 28.4	19.8 / 17.2 / 9.6
Qwen3-4B	Base	64.0 / 50.0 / 26.0	94.3 / 99.1 / 46.7	81.3 / 72.0 / 49.5	28.6 / 31.8 / 16.0	14.3 / 10.7 / 5.6
Qwen3-4B	CCOPD	74.0 / 53.0 / 38.0	95.2 / 99.1 / 81.9	81.3 / 74.8 / 70.1	27.0 / 25.1 / 22.4	13.2 / 9.8 / 10.4
Llama3.1-8B	Base	39.0 / 32.0 / 17.0	86.7 / 88.3 / 56.2	75.7 / 69.2 / 38.3	15.2 / 21.9 / 17.1	10.2 / 9.0 / 8.1
Llama3.1-8B	CCOPD	38.0 / 34.0 / 28.0	86.7 / 86.7 / **85.1**	75.7 / 70.1 / 43.9	16.6 / 19.4 / 17.3	12.5 / 6.5 / 8.3
Qwen3-8B: objective and teacher-source ablations						
Qwen3-8B	Forward-CCOPD	74.0 / 56.0 / 46.0	96.2 / 98.1 / 79.1	83.2 / 72.9 / 70.1	36.4 / 33.0 / 27.6	14.7 / 10.5 / 11.1
Qwen3-8B	4B teacher	75.0 / 62.0 / 46.0	97.1 / 98.1 / 80.0	84.1 / 75.7 / **75.7**	34.7 / 32.4 / 29.1	15.6 / 11.1 / 10.7
Qwen3-8B	14B teacher	70.0 / 59.0 / 46.0	98.1 / 99.1 / 84.8	85.1 / 75.7 / 67.3	37.8 / 34.4 / 27.1	14.8 / 10.3 / 12.1
Qwen3-8B: complete evaluation and baselines						
Qwen3-8B	Base	70.0 / 62.0 / 33.0	97.1 / 99.1 / 58.1	87.9 / 74.8 / 54.2	38.3 / 35.6 / 23.8	15.5 / 10.4 / 13.6
Qwen3-8B	SFT	75.0 / 60.0 / 39.0	98.1 / 98.1 / 84.8	86.9 / 72.0 / 68.2	36.0 / 31.4 / 23.7	16.0 / 10.6 / 12.8
Qwen3-8B	OPSD	77.0 / 55.0 / 35.0	98.1 / 99.1 / 67.6	87.9 / 78.5 / 60.8	38.7 / 36.1 / 24.1	13.2 / 7.7 / 9.4
Qwen3-8B	GRPO	75.0 / 75.0 / 33.0	97.1 / 99.1 / 54.3	87.9 / 77.6 / 49.5	38.0 / 35.6 / 24.1	15.9 / 10.9 / 12.8
Qwen3-8B	CCOPD	75.0 / 60.0 / **49.0**	98.1 / 99.1 / 83.8	85.1 / 73.8 / 70.1	37.3 / 32.9 / **29.4**	15.6 / 10.7 / **14.5**

Table 4: OOD task details. Each cell reports FULL/CONCAT/RAW-SHARDED (F/C/R).

level KL gives

$$D_{\mathrm{KL}}(P_\theta^h \| P_0^c) = \sum_{y \in \mathcal{A}} P_\theta^h(y) \log \frac{P_\theta^h(y)}{P_0^c(y)} = \mathbb{E}_{y \sim P_\theta^h} \left[\sum_{t=1}^{\tau(y)} \log \frac{\pi_\theta(y_t \mid h, y_{<t})}{\pi_0(y_t \mid c, y_{<t})} \right]. \tag{15}$$

Conditioning on a realized prefix $y_{<t}$ under P_θ^h, the next token y_t is distributed as $\pi_\theta(\cdot \mid h, y_{<t})$. Therefore,

$$\mathbb{E}\left[\log \frac{\pi_\theta(y_t \mid h, y_{<t})}{\pi_0(y_t \mid c, y_{<t})} \middle| y_{<t} \right] = D_{\mathrm{KL}}(\pi_\theta(\cdot \mid h, y_{<t}) \| \pi_0(\cdot \mid c, y_{<t})). \tag{16}$$

Applying Equation (16) inside Equation (15) proves Equation (11). □

By the definition of canonical deviation in the main text, $d_t(y) = \Psi_\theta(q, y_{<t})$. Thus, the token-level KL minimized by CCOPD is the on-policy decomposition of the sequence-level divergence between the history-conditioned student and the canonical FULL teacher.

Proposition 1 (Answer-event control)**.** *For any answer event $E \subseteq \mathcal{A}$, if $D_{\mathrm{KL}}(P_\theta^h \| P_0^c) \le \epsilon$, then*

$$\left| P_\theta^h(E) - P_0^c(E) \right| \le \sqrt{\epsilon/2}. \tag{17}$$

Proof. Pinsker's inequality gives

$$\mathrm{TV}(P_\theta^h, P_0^c) \le \sqrt{\frac{1}{2} D_{\mathrm{KL}}(P_\theta^h \| P_0^c)}. \tag{18}$$

For any event E, absolute probability difference is bounded by total variation:

$$\left| P_\theta^h(E) - P_0^c(E) \right| \le \mathrm{TV}(P_\theta^h, P_0^c). \tag{19}$$

Combining the two inequalities proves Equation (17). □

Relation to the implemented objective. The chain-rule argument above describes an ideal stochastic objective over complete final-answer strings. The implemented objective in Equation (10) is a practical surrogate: it uses a finite number of student rollouts, applies the loss only to final-answer positions $T_{\mathrm{ans}}(\hat{y})$, normalizes by answer length.

A.2 Pair Construction, Teacher Screening, and Leakage Blocking

The pair construction enforces task equivalence at the final answer state. The canonical side contains $c(q) = \mathrm{FULL}(q)$. The student side contains the retained multi-turn history ending at the final user turn. A RAW-SHARDED example is kept only if metadata verifies that all user shards have been revealed, that the retained history ends with a user message, and that teacher supervision applies only to the next assistant rollout after that final user turn. Examples failing these structural checks are removed before training.

CCOPD also blocks full-prompt leakage into the student path. The student prompt is exactly the retained sharded history. The canonical FULL prompt is used only inside the teacher forward pass,

with the adapter disabled. Teacher answer text is never inserted into the student target sequence; the teacher only provides next-token probabilities on the same answer prefix generated by the student.

A.3 Practical Implementation Notes

The set $T_{\text{ans}}(\hat{y})$ contains only positions belonging to the final assistant answer after the last user turn. The loss is normalized by $|T_{\text{ans}}(\hat{y})|$ so that longer answers do not automatically dominate the objective. In the primary math setting, decoding stops at EOS or at the task-specific numeric answer marker.

B Raw-Shard Evaluation Protocol

We evaluate long-context instruction following with a sequential raw-shard protocol. Each example is pre-partitioned into an ordered sequence of shards. At test time, the user reveals these shards one at a time in the fixed dataset order. After each newly revealed shard, the model produces a response under the full conversation history, but no answer is scored until the final shard has been revealed. This protocol keeps the information stream identical across models while allowing the model to revise or commit to intermediate answers as additional evidence arrives.We use a unified evaluation benchmark across all models, consisting of 105 function-calling, 100 code, 120 table-to-text, 107 text-to-SQL, 103 math, and 92 summarization instances.

For each example, the conversation starts with the task-specific system prompt. The system prompt contains only stable task context, such as available function schemas for action prediction or database schemas for text-to-SQL. It does not include future user shards. We run the assistant with deterministic decoding temperature and do not impose a request-level output-token cap in the raw-shard evaluation. Once the final shard has been processed, a single final answer is extracted from the last assistant response. For free-form response tasks this extraction is the identity function; for code it uses a task-specific code extractor; and for structured-answer tasks we use the configured system extractor, `gpt-4o-mini`. The extracted answer is then passed to the official task evaluator.

All models are evaluated with the same sharded examples, shard order, answer extraction timing, and scoring scripts. Failed generations or evaluator errors are counted as incorrect examples for accuracy-based tasks and as zero-score examples for score-based tasks. This convention keeps the reported number tied to the end-to-end ability to complete the raw-shard interaction, rather than to the subset of examples for which generation succeeds.Detailed scoring rules for each task family are provided in Table 5.

C Diagnostic Probe Details

Self-Anchor Attention Ratio. For each analyzed final-answer state, we define two token groups: G_{usr} contains tokens corresponding to completed task-relevant user evidence, and G_{self} contains earlier assistant tokens that express self-generated commitments, such as unsupported guesses, premature answers, or task-specific conclusions. Let $\mathcal{T}$ denote answer-token query positions and let R be the number of attention heads. For attention weight $A_{\ell,r,t,j}$ from query position t to prompt token j in head r of layer ℓ, the attention density for group $g \in \{\text{usr}, \text{self}\}$ is

$$D_\ell(g) = \frac{1}{|\mathcal{T}|\, R\, |G_g|} \sum_{t \in \mathcal{T}} \sum_{r=1}^{R} \sum_{j \in G_g} A_{\ell,r,t,j}. \quad (20)$$

The Self-Anchor Attention Ratio is

$$\text{SAAR}_\ell = \log \frac{D_\ell(\text{usr}) + \epsilon}{D_\ell(\text{self}) + \epsilon}, \quad (21)$$

where $\epsilon > 0$ is a smoothing constant. SAAR is used as a descriptive probe of final-answer attention allocation, not as a causal explanation of model reasoning.

Removing process-reply commitments For each completed RAW-SHARDED history, we extract a process-stage wrong answer z_{self} from earlier assistant replies: unsupported numeric commitments or premature answer statements that are not entailed by the canonical user evidence. We then compute the final-state gold-vs-anchor margin

$$m_{\text{raw}} = \log p(z^\star \mid H_{\text{raw}}) - \log p(z_{\text{self}} \mid H_{\text{raw}}), \quad (22)$$

using length-normalized answer log-probabilities after the same answer prefix. We call an example wrong-anchor anchored when $m_{\text{raw}} \leq 0$, i.e., the base model assigns at least as much probability to its own earlier wrong commitment as to the gold answer. Masking the process-reply commitment spans changes this margin by

$$\Delta m_{\text{self}} = m_{\text{masked self}} - m_{\text{raw}}. \quad (23)$$

Domain	Benchmark source	Raw-shard interaction	Scoring rule
Math	GSM8K-style grade-school math	The user provides the problem statement in ordered shards. The model may reason after each shard, but only the final response is considered for scoring.	The final numeric answer is extracted and normalized by removing superficial formatting such as commas, currency symbols, and terminal punctuation. Accuracy is exact match against the gold numerical answer.
Actions	Berkeley Function-Calling Leaderboard (BFCL)	The system prompt gives the available function schema. The user instruction is then revealed as raw shards. The model must output the final function-call sequence after observing all shards.	The full final response is parsed into an abstract syntax tree and checked against the reference function call with the BFCL AST checker. We report binary accuracy.
Code	HumanEval and LiveCodeBench-style Python generation	The user reveals the programming problem incrementally, including the relevant specification and starter-code information when present. The model's final answer should contain executable Python code.	A task-specific extractor selects the final Python function or class solution. The solution is renamed when needed to match the target signature and is executed against the released test cases. We report pass@1 as binary accuracy.
Database	Spider text-to-SQL	The database schema is placed in the system prompt. The natural-language query is provided through ordered raw shards, and the final response is expected to contain a complete SQL query.	The SQL query is extracted from the final response, normalized for whitespace, and evaluated by Spider execution match against the reference SQL on the target database. We report binary accuracy.
Data-to-text	ToTTo-style table description	The user provides table facts and contextual hints in ordered shards. The model generates a one-sentence description after each turn, with only the final description scored.	The final response is scored with SacreBLEU against the available reference descriptions. We report the corpus BLEU score on the 0–100 scale.
Summary	Summary of a Haystack	The first turn gives the summarization task and the first batch of documents. Each subsequent shard reveals additional documents and asks the model to rewrite the summary considering all documents seen so far.	The final summary is evaluated for insight coverage and citation quality using the configured summary evaluator, `gpt-4o-mini`. We report the joint summary score on the 0–100 scale.

Table 5: Evaluation details for the six raw-shard domains. The model receives shards sequentially and is scored only after the final shard. Math, actions, code, and database are reported as accuracy; data-to-text and summary are reported as task-specific scalar scores.

The effect is selective: among 537 base examples, 324 are wrong-anchor anchored and 213 are gold-preferred. Masking self-commitment spans shifts wrong-anchor states toward gold by $+0.861$ margin on average, but has almost no effect on gold-preferred states ($+0.005$). The difference is $+0.856$ with a bootstrap 95% CI of approximately $[0.65, 1.06]$ (permutation $p < 10^{-4}$).

Neutral-placeholder contrast. For each analyzed prefix i, let C_i^{raw} be the realized context containing previous assistant replies, let C_i^{wait} be the same context after replacing those replies with neutral waiting text, and let C_i^{full} be the corresponding canonical FULL reference context. For model M and evaluated context C, define

$$\psi_i^M(C) = D_{\text{KL}}\Big(p_M(\cdot \mid C) \;\Big\|\; \pi_0(\cdot \mid C_i^{\text{full}})\Big). \tag{24}$$

The assistant-associated deviation is

$$\Delta\psi_i^M = \psi_i^M(C_i^{\text{raw}}) - \psi_i^M(C_i^{\text{wait}}). \tag{25}$$

Positive $\Delta\psi_i^M$ indicates that the realized assistant text moves the model farther from the canonical FULL reference than the neutral replacement does.

D Full-Context Strong-Pollution Evaluation

This appendix details two full-context pollution tests used to evaluate whether Qwen3-8B-CCOPD

resists strong misleading information inserted into otherwise complete problem contexts. In both tests, the original full problem is available to the model. The only intervention is an additional polluted statement that introduces a per-instance wrong numeric anchor. This setting differs from fragmented multi-turn tests: the model has enough task information, but must decide whether to trust the polluted statement.

D.1 Shared Evaluation Protocol

We evaluate on 103 held-out math problems. Each example provides a full-context trajectory H_{full} containing the original system instruction and the complete user problem. For each example with gold numeric answer y, we construct a wrong numeric anchor $\tilde{y}$ using the same deterministic function used by the evaluation script:

$$\tilde{y} = \texttt{wrong_numeric_anchor}(y). \qquad (26)$$

For integer answers, this is usually $y + 1$; for decimal answers, it is $y + 1.0$; for unparsable answers, the fallback anchor is a fixed non-gold value. Thus, the pollution is not a generic answer such as zero, but an example-specific near-answer that is often plausible.

Both Qwen3-8B and Qwen3-8B-CCOPD are evaluated on exactly the same polluted histories. We use deterministic decoding with temperature 1, a maximum generation length of 4096 tokens, and ten run per example. The metric is exact final-answer accuracy. We also report pairwise transitions: *fixed* denotes examples where the base model is wrong and Qwen3-8B-CCOPD is correct; *regressed* denotes the opposite.

D.2 Pollution 1: Assistant-Side Wrong Solution

Construction. This setting tests whether the model follows a polluted assistant history after the full user problem has already been shown. Starting from H_{full}, we append an assistant turn that claims the problem has already been solved and states the wrong anchor as the final answer. We then append a final user turn asking the model to provide the answer.

Polluted assistant turn:

```
I have already solved the full problem.
My checked calculation gives the final
numeric answer as ỹ.
#### ỹ
```

Final user turn:

```
Please provide the final answer to the
original problem now.
```

The full evaluated history is therefore

$$H_{\text{assist}} = H_{\text{full}} \oplus [a_{\text{wrong}}(\tilde{y})] \oplus [u_{\text{final}}], \qquad (27)$$

where $a_{\text{wrong}}(\tilde{y})$ is the polluted assistant message and u_{final} is the final user request. This is a strong pollution setting because the wrong value is not merely mentioned; it appears as a completed prior assistant solution in the conversation history.

D.3 Pollution 2: User-Side Wrong Answer Hint

Construction. This setting inserts the wrong anchor directly into the full user message. It is therefore a stronger user-side contamination than assistant-history pollution: the false hint appears in the same message as the problem itself. Starting from H_{full}, we modify only the final user message by appending the following hint:

```
Important hint from a previous verified
calculation: the final numeric answer is
ỹ. Use this value as the target answer if
it is at all consistent with the problem.
```

The evaluated history is

$$H_{\text{hint}} = [\, s_{\text{orig}},\ u_{\text{full}} \| h_{\text{wrong}}(\tilde{y}) \,], \qquad (28)$$

where s_{orig} is the original system message, u_{full} is the original complete problem, and $h_{\text{wrong}}(\tilde{y})$ is the appended false hint. The model must choose between solving the original problem and following a user-provided "verified" answer hint.

D.4 Interpretation

These two tests probe different sources of full-context contamination. In the assistant-side setting, the polluted answer is located in the conversation history as a prior assistant solution. The large improvement ($33.0\% \rightarrow 89.32\%$) suggests that CCOPD teaches the model to avoid blindly continuing from unreliable assistant-side state and to recompute from the user-provided evidence.

The user-side hint setting is harder in a different way because the pollution is attached to the user message itself. The base model still follows the false hint often enough to fall to 63.10%, whereas Qwen3-8B-CCOPD remains at 88.34%. This supports a stronger robustness claim than the earlier weak false-premise test: Qwen3-8B-CCOPD can

source	pairs	Math RAW-SHARDED
None	–	66.02%
HotpotQA	1k	76.69%

Table 6: Reciprocal source-domain check. Training CCOPD on non-math HotpotQA sharded histories still improves math RAW-SHARDED accuracy. This suggests that the useful signal is not tied to math-format supervision alone, but to aligning final answers under completed sharded evidence.

resist a per-example, near-gold, explicitly "verified" wrong-answer hint when the full problem is available.

The appropriate paper claim is mechanism-specific rather than universal: Qwen3-8B-CCOPD substantially improves robustness to full-context contamination when the pollution appears as assistant-side history or as a direct false-answer hint. This should not be overstated as robustness to every possible full-context pollution format.

D.5 Implementation Notes

The wrong anchor is generated deterministically from the gold answer before evaluation and is inserted identically for both models. No model-specific prompting, reranking, or post-hoc selection is used. Scoring is performed by an exact final-answer evaluator, so a response is counted as correct only when its extracted numeric answer matches the reference answer.

E Reciprocal Source-Domain Check: HotpotQA to Math

The main experiments train on math and evaluate transfer to non-math RAW-SHARDED tasks. Here we run the reverse check: we train on a non-math multi-hop QA source and evaluate on math RAW-SHARDED. We construct FULL/RAW-SHARDED pairs from HotpotQA train v1.1 (Yang et al., 2018), using the same final-answer alignment format as in the main setup: the student observes the raw sharded history, while the teacher is conditioned on the corresponding FULL context. The training data contains no math examples.

The HotpotQA run uses Qwen3-8B with LoRA rank 16 and alpha 32, learning rate 3×10^{-5}, 400 training steps, and the OPD-only full/RAW-SHARDED objective. Conservative scoring covers 103 completed math RAW-SHARDED task IDs, of which 79 are correct. We treat this as a source-domain sanity check rather than a full replacement for the six-domain main evaluation.

F Math Training Shard Construction

This appendix specifies how the math training shards are constructed, since the effect of CCOPD depends on the exact conversation history h. We separate two objects: the *static shard list*, which is deterministically derived from a math word problem, and the *raw training history*, which is the observed multi-turn interaction generated from that shard list.

Source pool and filtering. For the reported math run, we construct a 8k-example pool from GSM8K and GSM8K-Aug, with 4k examples from each source. Sampling uses a fixed seed (20260430). We discard questions longer than 1,200 characters, examples without a normalized final numeric answer, and examples that cannot be split into at least two shards. To reduce contamination with the internal math evaluation set, we exclude the 103 GSM8K indices used in the math held-out, train, pilot, and evaluation files. We also exclude exact normalized question-text overlaps with these held-out GSM8K questions, which covers GSM8K-Aug restatements when they are text-identical after whitespace and case normalization. In the resulting 8k-example pool, there are no normalized overlaps with the excluded questions. We do not perform semantic paraphrase-level filtering beyond this exact normalized-text check.

Deterministic shard boundaries. Given a question x, we first normalize whitespace and split x into sentence-like units using punctuation boundaries. If this yields fewer than two units, we fall back to splitting on common connective words such as `and`, `while`, `if`, `when`, `then`, and `but`. We then identify the query shard as the last unit containing a question mark; if no such unit exists, the final unit is used. The static shard sequence is

$$(s_1, s_2, \ldots, s_m) = (\text{query}, \text{remaining facts in original order}). \tag{29}$$

Thus the first user-side shard is intentionally under-specified: it asks for the desired quantity before all facts are available. The remaining shards reveal the arithmetic facts needed to solve the problem. We do not paraphrase or manually rewrite static shards at this stage; each shard is a contiguous segment of the original dataset question. This avoids introducing

a separate instruction-engineering variable into the shard design.

Raw training histories. For the full-raw-sharded CCOPD setting, the static shard list is converted into a raw interaction using the same sharded simulator for all examples. The user agent is constrained to reveal at most one shard per turn, reveal the entire selected shard, avoid repeating already revealed shards, and rephrase the shard in short conversational language. The simulator is run with Qwen3-8B as the assistant, user, and system-side local model. We keep a training pair only if every static shard is revealed at least once. The final assistant message, if present, is stripped so that the history ends at the final user turn:

$$h_{\text{raw}} = (m_{\text{sys}}, u_1, a_1, \ldots, u_m), \tag{30}$$

The target is the canonical full-question solution from the original example. Consequently, the learned behavior is not to answer each partial shard, but to produce the final solution only after all shards have been supplied. The same source records also define the full, concatenated, user-only sharded, and wait-trajectory variants; the raw CCOPD variant differs only in replacing the idealized user sequence with the observed raw sharded history.

Scope of the construction. This construction is designed to test whether a model can delay and revise reasoning as missing facts arrive, rather than whether it can follow a more carefully engineered prompt. The shard boundaries are deterministic and lightweight, so they are reproducible, but they are not intended to exhaust the space of natural multi-turn math conversations. User-side phrasing diversity in the raw histories comes from the constrained user simulator's rephrasings, while the underlying information units remain fixed. This is why evaluation is reported separately on the LIC release domains: it tests whether improvements from this math shard construction transfer beyond the exact GSM8K-style training distribution.

G Evaluator Robustness and Bootstrap Significance

Some tasks use LLM-based components for answer extraction or scoring. To check whether the reported RAW-SHARDED gains depend on these components, we compare the primary evaluator with a deterministic or restricted audit evaluator whenever such an audit is available. All comparisons are paired: Base and CCOPD are evaluated on the same RAW-SHARDED examples with the same shards, answer-extraction timing, and scoring scripts as in Table 1. We report the CCOPD–Base delta under both the primary evaluator and the audit evaluator, together with paired bootstrap confidence intervals over task examples.

Interpretation. The audit suggests that the main RAW-SHARDED gains are unlikely to be artifacts of LLM-based extraction or judging. For deterministic or restricted audit evaluators, the CCOPD–Base delta remains positive across all task families. Code, function calling, and ToTTo are fully deterministic under the reported evaluators, and their audit deltas exactly match the primary deltas. For math and text-to-SQL, the audit evaluators give larger deltas than the primary pipelines. This indicates that the primary LLM-assisted extraction is not systematically favoring CCOPD; if anything, it is conservative for the reported CCOPD–Base difference.

Math has lower primary–audit agreement than the other discrete tasks because the two extractors differ in how they handle answer formatting and final-number selection. However, the deterministic regex audit preserves the positive effect and yields a larger delta. The conclusion therefore does not depend on the LLM-based extractor.

SummHay is the weakest case. Both the primary joint evaluator and the coverage-only audit show a positive CCOPD–Base trend, but the confidence intervals include zero. We therefore treat SummHay as a positive but non-significant trend, rather than as a statistically established improvement. Overall, the evaluator audit supports the robustness of the main conclusion: the improvements on RAW-SHARDED are not driven by evaluator preference for CCOPD, and the strongest claims are supported by deterministic or restricted audit evaluations.

H Lightweight Test-Time Mode Details

We evaluate two lightweight test-time modes that leave the model weights unchanged and intervene only at inference time. The first is a reset-then-answer mode, which re-centers the model on the current task state before each response. The second is a defer-until-complete mode, which asks the model to check whether the information required for solving the task is already complete before producing a final answer. We avoid the shorthand

Example	Original question	Static shards
GSM8K, gold answer 1860	Jenny is planning her catering budget for her wedding. She is going to have 80 guests. 3 times as many guests want steak as chicken. If each steak entree costs $25 and each chicken entree costs $18, how much is the total catering budget?	S1: If each steak entree costs $25 and each chicken entree costs $18, how much is the total catering budget? S2: Jenny is planning her catering budget for her wedding. S3: She is going to have 80 guests. S4: 3 times as many guests want steak as chicken.
GSM8K-Aug, gold answer 90	Ben planned to swim 30 laps per day for 10 days. But after 3 days he strained his leg and had to stop. How many laps has he finished?	S1: How many laps has he finished? S2: Ben planned to swim 30 laps per day for 10 days. S3: But after 3 days he strained his leg and had to stop.

Table 7: Actual math training examples after static shard construction. The query is placed first, and supporting facts are supplied afterward in the original textual order.

Domain	Primary evaluator	Audit evaluator	Agr./corr. ↑	Bias gap → 0	Δ_{primary} ↑ [95% CI]	Δ_{audit} ↑ [95% CI]
Math	gpt-4o-mini extraction + numeric EM	regex numeric EM	83.5	-5.8	+16.5 [6.8, 26.2]	+22.3 [10.7, 34.0]
Code	deterministic extractor + tests	same	100.0	0.0	+16.0 [8.0, 25.0]	+16.0 [8.0, 25.0]
Function call	BFCL AST checker	regex calls + BFCL AST	100.0	0.0	+25.7 [16.2, 35.2]	+25.7 [16.2, 35.2]
Text-to-SQL	gpt-4o-mini SQL ext. + Spider exec.	regex SQL ext. + Spider exec.	95.8	-4.7	+15.9 [5.6, 26.2]	+20.6 [10.3, 30.8]
ToTTo	SacreBLEU	same	$r = 1.00$	0.0	+5.6 [2.8, 8.4]	+5.6 [2.8, 8.4]
SummHay	gpt-4o-mini joint judge	coverage-score variant	$r = 0.61$	-1.2	+0.9 [-1.3, 3.1]	+2.1 [-3.3, 7.7]

Table 8: Evaluator robustness and paired-bootstrap audit for Table 1. Δ denotes the CCOPD–Base difference on RAW-SHARDED under the corresponding evaluator. Agr. is output-level agreement for discrete tasks; corr. is Pearson correlation for scalar generation scores. Bias gap is $(E_{\text{primary}} - E_{\text{audit}})_{\text{CCOPD}} - (E_{\text{primary}} - E_{\text{audit}})_{\text{Base}}$. Values near zero indicate that the primary evaluator is not systematically more favorable to CCOPD than the audit evaluator. Negative values indicate that the primary evaluator is, if anything, more conservative for the CCOPD–Base delta. Confidence intervals are paired bootstrap intervals over task examples. For SummHay, the audit evaluator is the logged coverage-only score variant, which removes the citation component from the primary joint score without introducing another model call.

Model	Default	reset-then-answer	defer-until-complete
Base	66.0	72.8	67.0
CCOPD	**82.5**	79.6	77.7

Table 9: Accuracy (%) on math RAW-SHARDED (103 examples) under two lightweight test-time modes.

"WAIT" here to distinguish this test-time mode from the separate WAIT-replacement mechanism probe used in the main text.

The two modes show a clear asymmetry. For the base model, the reset-then-answer mode yields a meaningful gain (68/103 → 75/103), whereas the defer-until-complete mode yields only a marginal improvement (68/103 → 69/103). This pattern suggests that the raw-sharded failure is not simply premature answering; explicitly re-centering the current task state is more helpful than merely encouraging deferral. Direct CCOPD remains strongest under the default setting (85/103). Applying the same modes to the trained model lowers accuracy slightly (82/103 and 80/103), suggesting that part of the relevant task-state management behavior has already been internalized and that the extra meta-instructions are redundant or mildly disruptive.

Reset-then-answer mode.

> Before every response, explicitly summarize the current task goal in one short sentence under "Current goal:". Then continue with your response. When enough information is available, provide the final answer in the task's required format.

Defer-until-complete mode.

> Before every response, explicitly check whether all necessary conditions for solving the task are available under "Condition check:". If any necessary condition is missing or ambiguous, do not produce a final answer yet; state what information is still missing and wait for more information. Only after the conditions are complete, provide the final answer in the task's required format.

I Potential Risks and Ethical Considerations

This work aims to improve the reliability of language models in completed multi-turn interactions

by reducing sensitivity to self-generated assumptions. The method is not designed for any specific high-stakes deployment and does not remove the need for existing safety, privacy, or human-oversight mechanisms. A possible dual-use risk is that more stable multi-turn instruction following could also make harmful workflows more reliable, including misleading content generation, social-engineering conversations, or unsafe tool-use pipelines. In addition, improved robustness under contaminated histories may increase user trust in final answers even when the model is still wrong, especially in domains not covered by our evaluation. Our experiments are limited to the studied task families and primarily evaluate task correctness rather than fairness, privacy, or security behavior; therefore, CCOPD should not be interpreted as a general safety or factuality guarantee. We recommend that deployments combine this type of training with domain-specific safety filters, privacy-preserving data handling for conversational histories, red-team evaluation, and human review in high-stakes settings.

J Training Details

Table 10 shows training details

Besides, We report the main external packages used for model training, model serving, answer extraction, and evaluation. Table 11 is from the environment used for the reported experiments. We do not use NLTK, SpaCy, or ROUGE for the primary reported metrics.

K AI Assistant Use Disclosure

The authors used ChatGPT as an AI assistant for limited coding support and language editing, including drafting and debugging experimental scripts, improving grammar and clarity, and suggesting alternative phrasing. All scientific ideas, experimental design, results, claims, citations, and final text were reviewed, verified, and edited by the human authors. ChatGPT was not used to generate or fabricate experimental results, make autonomous research decisions, or satisfy authorship criteria, and no AI system is listed as an author.

L Artifact Licenses, Terms, and Intended Use

Artifact Licenses and Terms of Use We use publicly available model checkpoints, benchmark datasets, and evaluation code only for research purposes and follow the terms specified by their original releases. Qwen3 models are released under the Apache 2.0 license, while Llama3.1 models are governed by the Meta Llama 3.1 Community License. The evaluation and training artifacts used in this work include GSM8K, HumanEval, LiveCodeBench, Berkeley Function-Calling Leaderboard (BFCL), Spider, ToTTo, Summary of a Haystack (SummHay), HotpotQA, and the Lost in Conversation sharded-instruction release. Their licenses or release terms include MIT, Apache 2.0, CC BY-SA 4.0, Creative Commons Share-Alike 3.0, and model-specific community licenses, depending on the artifact.

Our derived artifacts consist of transformation scripts, sharded conversation histories, retained canonical/history pairs, evaluation logs, and LoRA adapters. If released, code and scripts will be distributed under an open-source software license, and derived data will be released only when permitted by the corresponding upstream licenses. For datasets with share-alike requirements, we will preserve attribution and distribute derived data under compatible terms, or release only reconstruction scripts and example identifiers when redistribution of transformed data is not appropriate. Model adapters, if released, will be distributed subject to the license of the corresponding base model and the terms of the training data. We do not redistribute proprietary API models or closed-source model outputs as standalone model artifacts.

Artifact Use and Intended Use All external artifacts are used for research purposes consistent with their released benchmark or model-development roles. We use pretrained models only as base models, frozen teachers, or research baselines, and use public datasets and evaluation suites only to study math reasoning, code generation, function calling, text-to-SQL, table-to-text generation, summarization, and multi-turn instruction following. The derived artifacts created in this work, including sharded histories, canonical/history pairs, transformation scripts, logs, and LoRA adapters, are intended only for research on multi-turn language-model reliability. If released, derived data and adapters will follow the licenses and access conditions of the corresponding upstream artifacts; when redistribution of transformed examples is not clearly permitted, we will release only reconstruction scripts and example identifiers. We do not

Item	Configuration
Model	Qwen3-8B with LoRA fine-tuning
Variant	Qwen3-8B CCOPD, 6K-data setting
Training data	6,000 raw-sharded math conversations from GSM8K/GSM8K-Aug
Training objective	CCOPD KL-only objective
Teacher signal	Same Qwen3-8B backbone conditioned on the full problem context
Student context	Raw sharded conversation ending at the final user turn
Precision	bfloat16; no 4-bit quantization
Maximum sequence length	8,192 tokens
Rollout budget	4,096 new tokens
Rollout decoding	temperature 1.0, top-p 0.95
LoRA rank / alpha / dropout	16 / 32 / 0.05
LoRA target modules	query, key, value, output, gate, up, and down projections
Trainable parameters	43.65M, corresponding to 0.53% of the backbone
Optimizer	AdamW
Learning rate	3×10^{-5}
Warmup	0.03 ratio, corresponding to 90 warmup steps
Batch size	8 conversations
Gradient accumulation	1 step
Effective batch size	8 conversations
Epochs	4
Optimizer updates	3,000
Random seed	20260429
Training compute	≈132 GPU-hours on NVIDIA GeForce RTX 4090 (24 GB GDDR6X)

Table 10: Training configuration for the CCOPD setting.

collect private user conversations or new human-subject data, and we do not attempt to identify individuals or infer private attributes from benchmark examples.

M Artifact Documentation

This work uses existing public benchmarks and derived sharded presentations for evaluating multi-turn robustness. All language data used in our experiments is English. The training source for the main CCOPD run is restricted to math word-problem examples from GSM8K and GSM8K-Aug. The evaluation suite covers six task families: grade-school math word problems, Python code generation, function calling, text-to-SQL, table-to-text generation, and long-context summarization. We additionally use HotpotQA only for a reciprocal source-domain check.

Domains and task coverage. The math domain consists of grade-school arithmetic word problems. The code domain consists of Python programming tasks from HumanEval- and LiveCodeBench-style evaluations. The function-calling domain uses BFCL-style tool-use tasks with function schemas. The database domain uses Spider-style text-to-SQL tasks with database schemas. The data-to-text domain uses ToTTo-style table description tasks. The summarization domain uses Summary-of-a-Haystack-style long-context summarization. These domains are used to test whether a model trained only on sharded math conversations transfers to non-math RAW-SHARDED histories.

Constructed sharded histories. For each task instance, the original task information is converted into task-equivalent presentation modes. FULL presents the complete task in one prompt, CONCAT concatenates all user-provided shards, and RAW-SHARDED reveals the same user evidence incrementally through a multi-turn interaction. In the main training setting, the student sees only the retained RAW-SHARDED history ending at the final user turn, while the frozen teacher sees the corresponding FULL prompt. We retain only examples for which metadata verifies that all task-relevant user shards have been revealed before the final-answer state.

Linguistic and interaction phenomena. The artifacts are designed to study multi-turn instruction following under incremental evidence release. The main phenomena include delayed constraint integration, revision after partial information, sensitivity to earlier assistant replies, and robustness to self-generated commitments in the conversation history. The benchmark is not intended to represent all natural multi-turn dialogue patterns; it focuses on task-equivalent histories where the final user evidence is complete.

Component	Version	Usage
Python	3.10.18	Runtime environment for training, serving, and evaluation scripts.
PyTorch	2.6.0+cu124	Model training and inference; experiments use bfloat16 precision.
Transformers	4.57.1	Loading Qwen models and tokenizers via HuggingFace APIs and chat templates.
PEFT	0.17.1	LoRA adapter construction, loading, and saving.
Accelerate	1.10.1	Distributed and device-management utilities for multi-GPU runs.
Tokenizers / SentencePiece	0.22.1 / 0.2.1	Tokenization backends used by the HuggingFace Qwen tokenizer.
Safetensors	0.6.2	Serialization format for LoRA adapter weights.
OpenAI Python SDK	2.2.0	OpenAI-compatible client for local model endpoints and GPT-based answer extraction/judging.
FastAPI / Uvicorn / Pydantic	0.118.0 / 0.37.0 / 2.11.10	Local OpenAI-compatible HTTP server for HuggingFace model inference.
SacreBLEU	2.6.0	ToTTo/data-to-text evaluation using `corpus_bleu`; scores are divided by 100.
NumPy / Pandas	2.2.6 / 2.3.3	Aggregation and post-processing of evaluation results.

Table 11: Key package versions used in the reported experiments.

Demographic information. We do not collect new human-subject data and do not infer or annotate demographic attributes. The public source benchmarks used in this work do not provide systematic demographic annotations for authors or subjects of the examples. Accordingly, our experiments do not support claims about demographic-group coverage or demographic fairness.

Evaluation artifacts. All models are evaluated on the same sharded examples, shard order, answer-extraction timing, and scoring scripts. Math, code, function calling, and text-to-SQL are evaluated with accuracy-style metrics; table-to-text and summarization use task-specific scalar scores. For tasks involving LLM-assisted extraction or judging, we additionally report deterministic or restricted audit evaluators when available.