ACC: COMPILING AGENT TRAJECTORIES FOR LONG-CONTEXT TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent development of agents has renewed demand for long-context reasoning capacity of LLMs. However, training LLMs for this capacity requires costly long-document curation or heuristic context synthesis. We observe that agents produce massive trajectories when solving problems, invoking tools and receiving environment observations across many turns. The evidence needed to answer the original question is thus scattered throughout these turns, requiring integration of distant context segments. Nevertheless, standard agent SFT masks tool responses and only trains turn-level tool selection, creating a supervision blind spot where these scattered signals go unused. We propose Agent Context Compilation (ACC), which converts trajectories from search, software engineering, and database querying agents into long-context QA pairs that combine the original question with tool responses and environment observations gathered across multiple turns, training the model to answer directly without tool use. This makes the dependencies between the question and the evidence explicit, enabling direct supervision of long-context reasoning over distant segments without additional annotation. ACC is a simple but effective approach that can be combined with any existing long-context extension or training method, providing scalable supervised fine-tuning data. We validate ACC on long-range dependency modeling tasks through MRCR and GraphWalks, challenging benchmarks requiring cross-turn coreference resolution and graph traversal over extended contexts. Training Qwen3-30B-A3B with ACC achieves 68.0 on MRCR (+17.8) and 76.4 on GraphWalks (+6.5), results comparable to Qwen3-235B-A22B, while preserving general capabilities on GPQA, MMLU-Pro, AIME, and IFEval. Further mechanism analysis reveals that the ACC-trained model exhibits task-adaptive attention restructuring and expert specialization. Dataset and checkpoints will be released publicly.

1 INTRODUCTION

Recently, the rise of agents has brought fresh attention to long-context reasoning for LLMsOpenAI (2026); Anthropic (2026); Google DeepMind (2026); Qwen Team (2026), since agents work through many turns of tool calls and models need to handle increasingly long inputs. However, conventional training of LLMs for this capacity relies on costly long-document curation or heuristic context synthesis. Curating annotated long documents requires precise evidence labeling and intensive quality filtering. Heuristic synthesis gathers contexts without the complex dependencies that actual problem solving creates. These limitations severely restrict scalable training for long-span reasoning and motivate the exploration of alternative supervision sources.

Agents produce massive multi-turn trajectories when solving problems, invoking tools and receiving tool responses across many turns. The evidence needed to answer the original question is scattered throughout these turns, requiring integration of distant context segments. Although these trajectories can be directly used

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

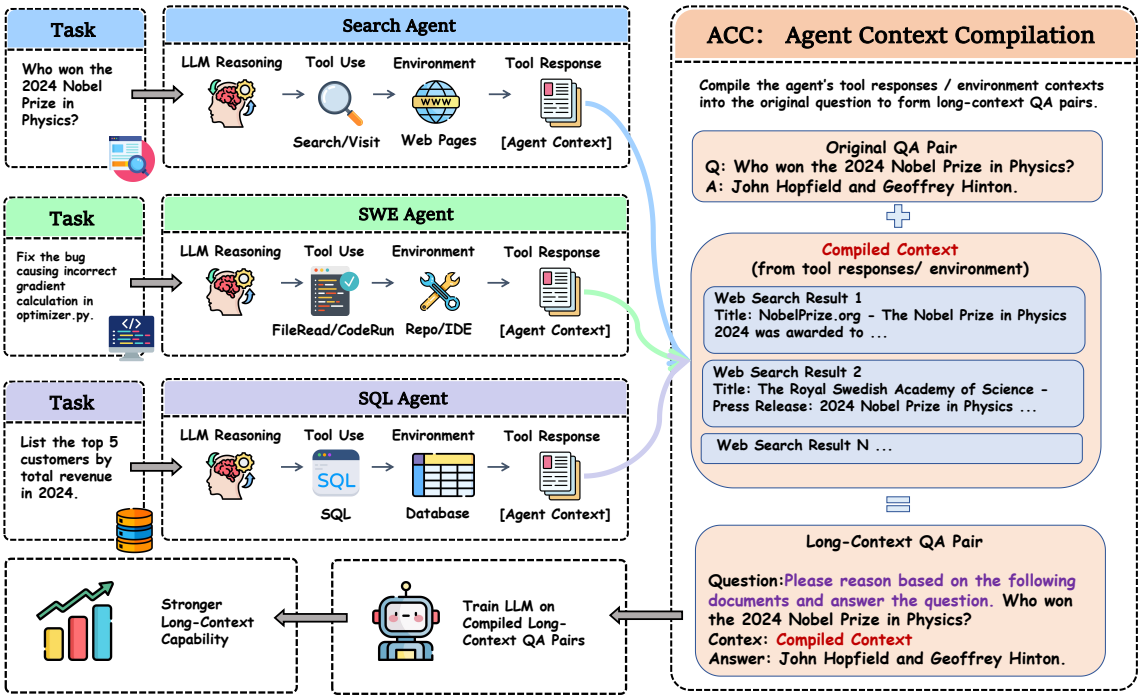


Figure 1: **Overview of ACC.** Multi-turn agent trajectories (Search, SWE, SQL) are compiled into long-context QA pairs by assembling tool responses and environment contexts.

for supervised fine-tuning, standard practice masks out tool responses and only supervises turn-level tool selection. This creates a supervision blind spot that leaves scattered evidence signals unused and severely limits the development of long-context capabilities.

To address this, we propose **Agent Context Compilation (ACC)**, which converts agent trajectories into long-context training data without additional human annotation. By assembling the original question with tool responses and environment observations gathered across multiple turns into one context, ACC makes the dependencies between the question and scattered evidence explicit, enabling direct supervision of long-context reasoning without additional annotation. **ACC is a simple but effective approach that can be combined with existing long-context extension or training methods, providing scalable supervised fine-tuning data.** Figure 1 illustrates the ACC pipeline.

We apply ACC to three representative agent classes including search agents that retrieve web pages to answer complex questions, SWE agents that inspect source files to resolve issues, and SQL agents that query relational tables for structured analytics. In each case, we compile **answer-verified trajectories** into long-context training pairs, taking the answer directly from the final output without additional human annotation.

We validate ACC on long-range dependency modeling tasks through MRCR and GraphWalksOpenAI (2025), two particularly challenging benchmarks requiring cross-turn coreference resolution and graph traversal over extended contexts. Training Qwen3-30B-A3B with ACC achieves 68.0 on MRCR (+17.8) and 76.4 on GraphWalks (+6.5), results comparable to Qwen3-235B-A22B, while preserving general capabilities on GPQA, MMLU-Pro, AIME, and IFEval. Mechanism analysis further reveals that the ACC-trained model exhibits task-adaptive attention restructuring and expert specialization, reflecting flexible adaptation to distinct long-range reasoning demands.

Contributions. Our main contributions are summarized as follows. (1) We propose Agent Context Compilation (ACC), a method that converts multi-turn agent trajectories into long-context training QAs. (2) We show that the ACC-trained Qwen3-30B-A3B achieves results comparable to Qwen3-235B-A22B on long-range dependency modeling benchmarks including MRCR and GraphWalks, while preserving general capabilities. (3) Through mechanism analysis, we observe task-adaptive attention restructuring and expert specialization emerging after ACC training, suggesting that the acquired long-range capacity manifests as flexible, task-specific patterns.

2 Related Work

2.1 Long-Context Capacity Evaluation

Evaluating long-context capabilities has evolved significantly. Early benchmarks such as NIAH Kamradt (2023) tested surface-level retrieval by embedding specific facts within distractor text. RULER Hsieh et al. (2024) extended this with variable tracking, aggregation, and multi-hop reasoning tasks. LongBench Bai et al. (2025) introduced diverse real-world tasks including QA, summarization, and code understanding. However, performance on these benchmarks has largely saturated, as they mainly test localized retrieval or single-turn reasoning within long contexts. Classic benchmarks such as Musique Trivedi et al. (2022) and NarrativeQA Kočiský et al. (2017) further targeted multi-hop reasoning and long-document narrative understanding. More recently, OpenAI released MRCR (Multi-Round Coreference Resolution) and GraphWalks OpenAI (2025) as direct tests of long-range dependency modeling. By requiring cross-turn coreference resolution and graph traversal over extended contexts, they are substantially harder than prior single-turn or retrieval tasks, and have become standard benchmarks for mainstream large model releases.

2.2 Long-Context Extension and Training

Recent efforts to improve long-context capabilities generally fall into four categories. First, pre-training methods modify position embeddings or attention mechanisms. MrRoPe Tian et al. (2026) applies RoPE interpolation and NTK-aware frequency scaling to broaden the context window. ROPE++ Liu et al. (2025) reuses the discarded imaginary component of RoPE's complex form to build parallel attention heads for improved length extrapolation. Native Sparse Attention Yuan et al. (2025) and Mamba-3 Lahoti et al. (2026) reduce complexity through sparse and linear attention. Second, some works focus on constructing high-quality long documents for pre-training data. Longwanjuan Lv et al. (2024) filters texts by coherence, cohesion, and complexity. LiteLong Jia et al. (2025) leverages book taxonomies and multi-agent debate for corpora retrieval and concatenation. Quest Tang et al. (2024) predicts possible questions and clusters core keywords to stitch short documents. These methods synthesize long texts rather than post-training QA pairs. Third, post-training recipes combine synthetic data with RL. longRLVR Chen et al. (2026) generates QA pairs with precise evidence block annotations from long texts. LongPO Chen et al. (2025) extracts key short chunks to build short-long preference pairs and applies short-to-long KL constraints in DPO. LoongRL Wang et al. (2025) proposes KeyChain to insert irrelevant documents for hard long-context synthesis and stabilizes GRPO with rule rewards and no entropy term. Fourth, employ agent frameworks at inference time to manage long-context memory. QwenLong-L1.5 Shen et al. (2025) cleans multi-source documents, builds knowledge graphs, and applies AEPO for dynamic entropy control. MemAgent Yu et al. (2025) mixes irrelevant HotpotQA documents and uses Multi-Conv DAPO to decompose long questions into multi independent conversations with memory updates. Our work differs by using agent trajectories as a direct data source for long-context reasoning training, rather than modifying architectures, synthesizing pre-training documents, or relying on complex post-training RL pipelines.

3 METHOD

3.1 THE SUPERVISION BLIND SPOT OF AGENT SFT

Standard agent SFT masks all tool responses (observations) and only supervises turn-level reasoning and actions. The model therefore never learns to integrate evidence scattered across multiple turns.

An agent trajectory consists of $k-1$ interaction turns followed by a final answer turn

$$\tau = (q, (r_1, a_1, o_1), \ldots, (r_{k-1}, a_{k-1}, o_{k-1}), (r_k, y)),$$

where r_t is reasoning, a_t is action, o_t is tool response (observation), and (r_k, y) is the final reasoning-answer pair. The history up to turn t is $\mathcal{H}_{<t} = (r_1, a_1, o_1, \ldots, r_{t-1}, a_{t-1}, o_{t-1})$[1]. Tool responses are masked from the loss and only model-generated tokens are supervised.

Formally, the standard objective is

$$\mathcal{L}_{\text{agent}} = -\sum_{t=1}^{k} \sum_{j \in \mathcal{I}_t} \log P(\text{token}_j \mid \mathcal{H}_{<t}, \text{token}_{<j}), \tag{1}$$

where $\mathcal{I}_t = r_t \cup a_t$ for $t < k$ and $\mathcal{I}_k = r_k \cup y$.

Grouping Eq. equation 1 by turn reveals its structure

$$\mathcal{L}_{\text{agent}} = \underbrace{\sum_{t=1}^{k-1} \Big(- \sum_{j \in r_t \cup a_t} \log P(\text{token}_j \mid \mathcal{H}_{<t}, \text{token}_{<j}) \Big)}_{\text{local next-tool selection}} + \underbrace{\Big(- \sum_{j \in r_k \cup y} \log P(\text{token}_j \mid \mathcal{H}_{<k}, \text{token}_{<j}) \Big)}_{\text{final answer prediction}}. \tag{2}$$

The first $k-1$ terms supervise only local reasoning and tool selection at each turn. Consider a token in tool response o_t at turn $t < k$. Excluded from the loss, it receives gradients only indirectly through subsequent unmasked tokens. The dominant signal flows along a **short path** to the next action a_{t+1}, where o_t lies in the immediate context. Any gradient relevant to the final answer y must back-propagate through a **long chain** of intermediate turns to reach o_t, and is heavily weakened. Consequently, these intermediate turns act as a **supervision filter**, so o_t is updated primarily to support local action prediction, ignoring answer-relevant features unless they also serve local needs. This is the supervision blind spot of agent SFT.

3.2 AGENT CONTEXT COMPILATION

ACC solves this problem by gathering all evidence into one long context C and training the model to write a reasoning trace r and final answer y directly from the question q and context C. The new training objective is

$$\mathcal{L}_{\text{ACC}} = -\sum_{j \in r \cup y} \log P(\text{token}_j \mid q, C, \text{token}_{<j}). \tag{3}$$

Unlike Eq. equation 1, this objective contains no intermediate action terms, so the final answer supervision reaches every evidence token directly without being filtered through turn-level tool selection. The model therefore learns to integrate scattered evidence into a global answer instead of merely optimizing local next-tool selection.

[1] We present interleaved reasoning traces for clarity. Non-interleaved variants do not affect our conclusions.

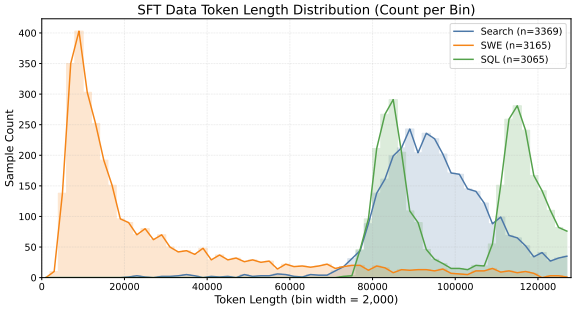


Figure 2: **Token length distribution of the ACC training data.**

Table 1: Training parameters for supervised fine-tuning.

Hyperparameter	**Value**
Sequence length	131,072 tokens
Global batch size	16
Learning rate	1×10^{-5} (min 1×10^{-6})
LR schedule	Cosine, 5% warmup
Optimizer	AdamW (β_1=0.9, β_2=0.999, decay 0.1)
Loss	Cross-entropy (chunk 1024)
Sequence parallelism	8
Training epochs	4

Given a set of **answer-verified trajectories** $\mathcal{T} = \{\tau_1, \ldots, \tau_N\}$, ACC converts each trajectory into a training example

$$\tau_i \mapsto (x_i, y_i, r_i),$$

producing a dataset $\mathcal{D} = \{(x_i, y_i, r_i)\}_{i=1}^{M}$. Here $x_i = (q_i, C_i)$ combines the original query with the compiled context, y_i is the final answer from the trajectory, and r_i is its reasoning trace.

3.3 Context Construction

For each trajectory we extract structured evidence pieces $\text{Evi}(\tau) = [e_1, \ldots, e_m]$ such that the aggregated context alone suffices to answer q without tool use. For **search trajectories** we extract the full text of visited pages and include unvisited candidate results as distractors. For **SWE trajectories** we extract files involved in the correct patch and include additional context files inspected during debugging as distractors. For **SQL trajectories** we extract the complete contents of all tables queried during the trajectory.

To increase task difficulty, we apply a random permutation π over $\{1, \ldots, m\}$ and concatenate the pieces into a compiled context

$$C_i = \text{Concat}(e_{\pi(1)}, e_{\pi(2)}, \ldots, e_{\pi(m)}), \quad \text{with } |C_i| \leq B, \tag{4}$$

where B is the token budget. Because evidence pieces are self-contained, shuffling forces the model to locate relevant information via semantic association rather than sequential position.

Answer-verified trajectories contain correct answers but lack explicit reasoning traces. We generate candidate rationales with **DeepSeek-V3.2-Thinking** and retain only those leading to the correct answer y_i via rejection sampling. For SWE, we add an extra filtering step where a **DeepSeek-V3.2-Thinking** judge verifies whether the rationale correctly derives the patch without hallucination. See Appendix F for the exact judge prompt. Pass rates after filtering are near 100% (Search), 50% (SQL), and 10% (SWE). The final example is (x_i, y_i, r_i) with $x_i = (q_i, C_i)$. A worked compilation example for each agent type is provided in Appendix A.

4 Experiments

4.1 Experimental Setup

Base Model. We use Qwen3-30B-A3B-Thinking Yang et al. (2025) as our base model.

Training Configuration. We compile 9,599 trajectories in total (Search: 3,369; SWE: 3,165; SQL: 3,065), with compiled context lengths ranging from 2K to 128K tokens and distinct per-agent length distributions (Figure 2). The details of training parameters are summarized in Table 1.

Table 2: Long-range dependency modeling benchmark results (avg@3). Numbers in parentheses show improvement over the Qwen3-30B-A3B-Thinking baseline. All results use default inference configurations without manual tuning of reasoning effort.

Model	MRCR[1]			GraphWalks[2]		
	2-needle	4-needle	Overall	Parents	BFS	Overall
Base Model						
Qwen3-30B-A3B-Thinking	61.84	38.41	50.19	71.19	68.47	69.92
Our Method						
Qwen3-30B-A3B-Thinking + ACC (Ours)	77.29 (+15.45)	58.51 (+20.10)	**67.95** (+17.76)	81.42 (+10.23)	70.61 (+2.14)	**76.37** (+6.45)
Strong Baselines						
Qwen3-235B-A22B-Thinking	74.98	59.96	67.51	78.53	74.45	76.63
GPT-OSS-120B	46.72	29.16	38.00[3]	75.92	61.82	69.34
DeepSeek-V3.2-Thinking	81.60	60.32	71.01	89.87	80.26	85.39
GPT-5.1-Thinking	73.75	47.93	60.91	81.41	29.41	57.14
GLM-4.6-Thinking	73.63	55.33	64.53	80.29	77.41	78.95
Kimi-K2-Thinking	68.01	47.96	58.05	84.34	75.17	80.04

[1]MRCR reports overall and sub-task (2-needle, 4-needle) scores, following the evaluation setting in QwenLong-L1.5 Shen et al. (2025).
[2]GraphWalks reports overall and sub-task (Parents, BFS) precision, following the evaluation setting in LongCat-Flash-Omni Team et al. (2025).
[3]GPT-OSS-120B was evaluated using the HuggingFace checkpoint via vLLM, and its lower MRCR scores largely result from frequent harmony-format parsing failures when processing multi-turn inputs.

Evaluation Benchmarks. We primarily evaluate on long-range dependency modeling benchmarks including MRCR OpenAI (2025) (multi-round coreference resolution) and GraphWalks OpenAI (2025) (graph traversal), which require tracking long-range relational dependencies across extended contexts. We also monitor general capabilities on GPQA-Diamond Rein et al. (2023), MMLU-Pro Wang et al. (2024), AIME AIME, and IFEval Zhou et al. (2023) to check for negative transfer.

4.2 Main Results

Table 2 presents our main results on long-range dependency modeling benchmarks. On MRCR, ACC improves both the 2-needle and 4-needle settings, yielding an overall score of 67.95 (+17.76). On GraphWalks, ACC improves both the Parents and BFS sub-tasks, yielding an overall precision of 76.37 (+6.45). These results are comparable to Qwen3-235B-A22B on these long-range dependency modeling benchmarks despite having nearly 8$\times$ fewer active parameters. For completeness, we also report results on additional long-context benchmarks in Appendix B.

4.3 General Capability Preservation

Long-context training often raises concerns about negative transfer to general capabilities. As shown in Table 3, our ACC-trained model achieves slight improvements on GPQA-Diamond (+2.49), MMLU-Pro (+1.50) and AIME'25(+3.33), while performance on AIME'24 and IFEval remains stable. These results suggest that ACC does not introduce noticeable degradation to general abilities.

To verify that these gains do not reflect test-set leakage, we compare the semantic distribution of training queries against benchmark questions. For each trajectory, we extract only the user question, stripping retrieved

Table 3: General capability evaluation (avg@3). No significant negative transfer is observed.

Model	GPQA-Diamond	MMLU-Pro	AIME'24	AIME'25	IFEval
Base Model					
Qwen3-30B-A3B-Thinking	67.71	74.50	90.00	86.67	86.69
Our Method					
Qwen3-30B-A3B-Thinking + ACC (Ours)	**70.20** (+2.49)	**76.00** (+1.50)	90.00 (0.00)	**90.00** (+3.33)	86.14 (−0.55)

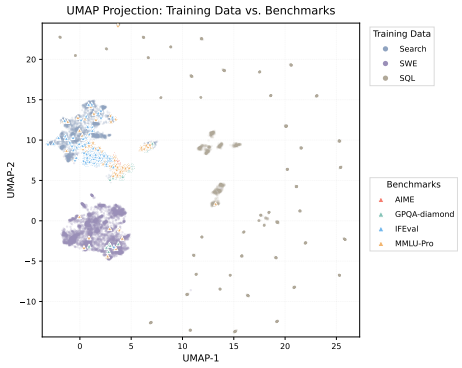


Figure 3: Two-dimensional UMAP projection of training queries (Search, SWE, SQL) and evaluation benchmark questions. Both training and evaluation samples are represented by their question text only.

Table 4: Quantitative separation between training queries and benchmark questions. Lower nearest-neighbor similarity and higher center distance both indicate limited overlap.

Benchmark	NN Sim.	Center Dist.
AIME	0.2832	0.8701
GPQA-Diamond	0.3557	0.7150
MMLU-Pro	0.3216	0.7685
IFEval	0.3425	0.9216
Overall AUC = 0.9986		

documents, code files, and database tables. Benchmark questions are similarly cleaned. Figure 3 shows the UMAP projection, and Table 4 reports quantitative metrics. Full details are in Appendix C.

The Search subset partially overlaps with general-knowledge benchmarks. Our multi-hop Search queries are synthesized from Wikipedia corpora, which naturally share topical vocabulary with knowledge benchmarks. The SWE and SQL subsets form distinct clusters. Quantitative analysis confirms this is domain-level overlap rather than instance duplication. The average nearest-neighbor cosine similarity remains below 0.36, and a linear classifier achieves an AUC of 0.9986 in separating training queries from benchmark questions. These patterns suggest the gains reflect transferable reasoning rather than data leakage.

4.4 Comparison with Long-Context Post-Training Methods

Table 5 compares ACC with recent long-context post-training methods. QwenLong-L1.5 Shen et al. (2025) leads on MRCR through a multi-stage pipeline involving document cleaning, knowledge-graph construction, and RL. ACC surpasses it on GraphWalks while requiring only standard SFT. LongPO Chen et al. (2025) and LongRLVR Chen et al. (2026) release models trained on the Qwen2.5 and are listed for reference.[2]

[2]LoongRL Wang et al. (2025) does not release trained checkpoints, so we do not include it in the comparison.

Table 5: Comparison with long-context post-training methods.

Model	MRCR	GraphWalks
Base Model		
Qwen3-30B-A3B-Thinking	50.19	69.92
Comparison Methods		
QwenLong-L1.5-30B[1]	92.30	73.85
Qwen2.5-7B-LongRLVR	19.76	15.72
Qwen2.5-14B-LongRLVR	20.06	22.78
Qwen2.5-7B-LongPO-128K[2]	31.50	12.97
+ ACC (Ours)	**67.95**	**76.37**

[1]QwenLong-L1.5 uses an agent framework that is not publicly available, so we evaluate it with standard inference within 256K.
[2]LongPO supports up to 128K context, and longer test instances are excluded.

Table 6: Agent-type and distractor ablations.

Training Data	MRCR	GraphWalks
Base Model		
Qwen3-30B-A3B-Thinking	50.19	69.92
Ablations		
+ Search (Agent SFT)	42.16 (-8.03)	57.87 (-12.05)
+ Search	58.33 (+8.14)	44.75 (-25.17)
+ Search (w/o distractor)	54.99 (+4.80)	58.46 (-11.46)
+ SWE	54.82 (+4.63)	50.66 (-19.26)
+ SWE (w/o distractor)	51.01 (+0.82)	52.88 (-17.04)
+ SQL	56.44 (+6.25)	75.50 (+5.58)
+ ACC (Ours)	**67.95** (+17.76)	**76.37** (+6.45)

4.5 Ablation Study

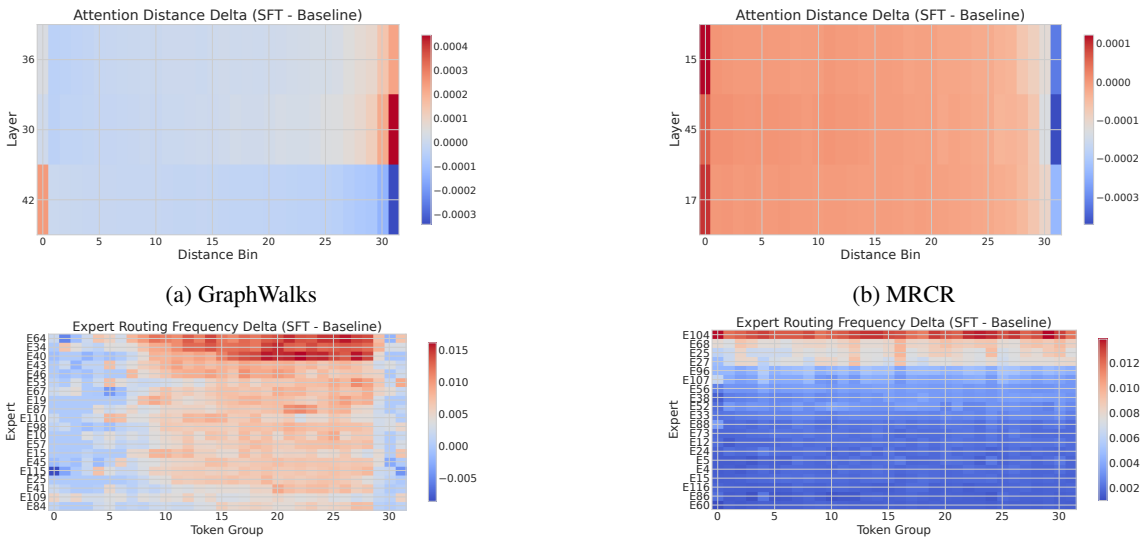
Agent-type ablation. Raw search trajectories with Agent SFT (observations masked) underperform the base model, confirming the **supervision blind spot** in Section 3.1. Table 6 ablates ACC by training on each agent type separately. All single-agent variants improve over the baseline on MRCR (Search +8.14, SWE +4.63, SQL +6.25), indicating that compiling scattered evidence into a single context alone improves cross-turn coreference resolution. On GraphWalks, however, only SQL improves (+5.58). This gap likely reflects evidence structure, since SQL tables are inherently relational and suit graph traversal, whereas web pages and source files are longer continuous passages that make node-level reasoning harder to learn. The full mixture surpasses all single-agent variants, showing that diverse trajectory types offer complementary coverage.

Distractor ablation. Removing distractors from Search and SWE lowers MRCR by 3.34 and 3.81 points, confirming that including unvisited results and unopened files helps the model learn to localize critical evidence. On GraphWalks, the trend reverses, with Search and SWE without distractors gaining +13.71 and +2.22. This is because these distractors are semantically unrelated to the query, helping noise filtering but offering little benefit for graph traversal. The full mixture, enriched by SQL's relational data, benefits from distractors for localization while preserving graph-walking capability, achieving the best overall result (76.37).

4.6 Mechanism Analysis

To understand how ACC improves long-range dependency modeling capacity, we visualize attention distance distributions and expert routing patterns on GraphWalks and MRCR examples.

Task-adaptive attention restructuring. Figure 4(a–b) shows attention distance distributions before and after ACC, with settings detailed in Appendix D. On GraphWalks, the ACC-trained model shows increased relative attention mass at both nearby and far-distance bins, consistent with the task requiring local neighborhood checks and distant node jumps. On MRCR, the model shows higher attention mass at nearby bins while preserving the baseline long-range profile, indicating improved precision in verifying candidate segments



(a) GraphWalks (b) MRCR

(c) GraphWalks (d) MRCR

Figure 4: Attention distance (top) and expert routing frequency (bottom) changes after ACC training (SFT minus baseline).

during scanning. Notably, the layers with the largest attention changes differ completely between the two tasks, suggesting the model adjusts its attention allocation per task rather than following a fixed pattern.

Expert specialization. Figure 4(c–d) shows changes in expert activation after ACC, with settings detailed in Appendix E. On GraphWalks, higher activation for distant token groups is distributed across several experts, suggesting balanced processing of cross-node jumps. On MRCR, one expert dominates across all token groups while most others are suppressed, pointing to dedicated processing of scanning and verification. The layers with the strongest activation shifts also differ completely across the two tasks, reflecting task-dependent expert specialization after ACC training.

5 Conclusion

We presented Agent Context Compilation (ACC), a simple method that compiles multi-turn agent trajectories into long-context training data and complements existing long-context training methods. The ACC-trained Qwen3-30B-A3B matches Qwen3-235B-A22B on MRCR and GraphWalks while largely preserving general capabilities, and mechanistic analyses reveal task-adaptive attention restructuring and expert specialization. Future work includes extending ACC to more agent types and longer contexts.

6 Limitations

ACC is evaluated on three agent types and one base model, and extending to more agent types and million-token contexts remains future work. The quality of the compiled rationales is further bounded by the teacher model, and robustness to the choice of teacher remains to be validated. Societal considerations, including privacy and intellectual property, are discussed in the Ethics Statement.

AI USE STATEMENT

We used generative AI tools only for language polishing of the manuscript and for assistance in writing analysis code; all such content was reviewed and verified by the authors. No generative AI tools were used for research ideation or experimental design. The use of a teacher model to generate candidate reasoning traces is part of the proposed data construction method, not authoring assistance. We take full responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work reuses existing agent trajectories and does not involve human subjects. Prior to release, we will filter the compiled dataset for private information and document data sources and licenses, as compiled contexts may contain copyrighted material. Reasoning traces in the dataset are teacher-generated and may propagate model biases; we retain only answer-verified traces and encourage downstream users to apply their own safety filtering.

REPRODUCIBILITY STATEMENT

We describe the data compilation procedure, training configuration, and evaluation protocols in the main text, and provide full details of all analyses in the appendix. The compiled dataset and trained checkpoints will be released publicly to enable replication of our results.

REFERENCES

AIME. American invitational mathematics examination. https://artofproblemsolving.com/wiki/index.php/AIME.

Anthropic. Claude opus 4.6 system card. https://www.anthropic.com/news/claude-opus-4-6, 2026. Accessed: 2026-05-02.

Yushi Bai, Shangqing Tu, Jiajie Zhang, Hao Peng, Xiaozhi Wang, Xin Lv, Shulin Cao, Jiazheng Xu, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. Longbench v2: Towards deeper understanding and reasoning on realistic long-context multitasks, 2025. URL https://arxiv.org/abs/2412.15204.

Guanzheng Chen, Xin Li, Michael Qizhe Shieh, and Lidong Bing. Longpo: Long context self-evolution of large language models through short-to-long preference optimization, 2025. URL https://arxiv.org/abs/2502.13922.

Guanzheng Chen, Michael Qizhe Shieh, and Lidong Bing. Longrlvr: Long-context reinforcement learning requires verifiable context rewards, 2026. URL https://arxiv.org/abs/2603.02146.

Google DeepMind. Gemini 3.1 pro. https://deepmind.google/models/gemini/pro/, 2026. Accessed: 2026-05-02.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. Ruler: What's the real context size of your long-context language models?, 2024. URL https://arxiv.org/abs/2404.06654.

Junlong Jia, Xing Wu, Chaochen Gao, Ziyang Chen, Zijia Lin, Zhongzhi Li, Weinong Wang, Haotian Xu, Donghui Jin, Debing Zhang, and Binghui Guo. Litelong: Resource-efficient long-context data synthesis for llms, 2025. URL https://arxiv.org/abs/2509.15568.

Greg Kamradt. Llmtest (needle in a haystack). https://github.com/gkamradt/LLMTest_NeedleInAHaystack, 2023. Accessed: 2026-05-02.

Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. The narrativeqa reading comprehension challenge, 2017. URL https://arxiv.org/abs/1712.07040.

Aakash Lahoti, Kevin Y. Li, Berlin Chen, Caitlin Wang, Aviv Bick, J. Zico Kolter, Tri Dao, and Albert Gu. Mamba-3: Improved sequence modeling using state space principles, 2026. URL https://arxiv.org/abs/2603.15569.

Xiaoran Liu, Yuerong Song, Zhigeng Liu, Zengfeng Huang, Qipeng Guo, Zhaoxiang Liu, Shiguo Lian, Ziwei He, and Xipeng Qiu. Beyond real: Imaginary extension of rotary position embeddings for long-context llms, 2025. URL https://arxiv.org/abs/2512.07525.

Kai Lv, Xiaoran Liu, Qipeng Guo, Hang Yan, Conghui He, Xipeng Qiu, and Dahua Lin. Longwanjuan: Towards systematic measurement for long text quality, 2024. URL https://arxiv.org/abs/2402.13583.

OpenAI. Introducing GPT-4.1, 2025. URL https://openai.com/index/gpt-4-1/. Accessed: 2026-05-02. Introduces the MRCR and GraphWalks benchmarks.

OpenAI. Gpt-5.4. https://openai.com/index/introducing-gpt-5-4/, 2026. Accessed: 2026-05-02.

Qwen Team. Qwen3.5. https://qwen.ai/blog?id=qwen3.5, 2026. Accessed: 2026-05-02.

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 qa benchmark, 2023. URL https://arxiv.org/abs/2311.12022.

Weizhou Shen, Ziyi Yang, Chenliang Li, Zhiyuan Lu, Miao Peng, Huashan Sun, Yingcheng Shi, Shengyi Liao, Shaopeng Lai, Bo Zhang, Dayiheng Liu, Fei Huang, Jingren Zhou, and Ming Yan. Qwenlong-l1.5: Post-training recipe for long-context reasoning and memory management, 2025. URL https://arxiv.org/abs/2512.12967.

Jiaming Tang, Yilong Zhao, Kan Zhu, Guangxuan Xiao, Baris Kasikci, and Song Han. Quest: Query-aware sparsity for efficient long-context llm inference, 2024. URL https://arxiv.org/abs/2406.10774.

Meituan LongCat Team, Bairui Wang, Bayan, Bin Xiao, Bo Zhang, Bolin Rong, Borun Chen, Chang Wan, Chao Zhang, Chen Huang, Chen Chen, Chen Chen, Chengxu Yang, Chengzuo Yang, Cong Han, Dandan Peng, Delian Ruan, Detai Xin, Disong Wang, Dongchao Yang, Fanfan Liu, Fengjiao Chen, Fengyu Yang, Gan Dong, Gang Huang, Gang Xu, Guanglu Wan, Guoqiang Tan, Guoqiao Yu, Haibo Qiu, Hao Lu, Hongbo Liu, Hongyu Xiang, Jiaheng Wu, Jian Yang, Jiaxing Liu, Jing Huang, Jingang Wang, Jinrui Ding, Juchao Jiang, Jun Kuang, Jun Wang, Junhui Mei, Ke Ding, Kefeng Zhang, Lei Chen, Liang Shi, Limeng Qiao, Liming Zheng, Lin Ma, Liuyang Guo, Liya Ma, Luying Sun, Man Gao, Mengshen Zhu, Miao Cao, Minliang Lin, Nuo Xu, Peng Shi, Qi Zhang, Qian Fang, Qian Wang, Qian Yang, Quanxiu Wang, Rongxiang Weng, Rongxin Guo, Ruoxuan Liang, Senbin Yang, Shanbo Xu, Shanglin Lei, Shengze Ye, Shimin Chen, Shuaiqi Chen, Shujie Hu, Shuo Li, Siqi Yang, Siyu Xu, Siyu Ren, Song Li, Songxiang Liu, Tianhao Bai, Tianye Dai, Wei Hong, Wei Wang, Weixiao Zhao, Wengang Cao, Wenlong Zhu, Wenlong He, Xi Su, Xi Nan, Xiaohan Zhao, Xiaohao Wang, Xiaoyu Zhao, Xiaoyu Wang, Xiaoyu Li, Xin Pan, Xin Chen, Xiusong Sun, Xu Xiang, Xudong Xing, Xuezhi Cao, Xunliang Cai, Yang Yang, Yanli Tan, Yao Yao, Yerui Sun, Yi Chen, Yifan Lu, Yin Gong, Yining Zhang, Yitian Chen, Yiyang Gan, Yuchen Tang,

Yuchen Xie, Yueqian Wang, Yuewen Zheng, Yufei Zhang, Yufeng Zhong, Yulei Qian, Yuqi Peng, Yuqian Li, Yuwei Jiang, Zeyang Hu, Zheng Zhang, Zhengkun Tian, Zhiqing Hong, Zhixiong Zeng, Zhuqi Mi, Ziran Li, Ziwen Wang, Ziyi Zhao, Ziyuan Zhuang, and Zizhe Zhao. Longcat-flash-omni technical report, 2025. URL https://arxiv.org/abs/2511.00279.

Qingyuan Tian, Wenhong Zhu, Xiaoran Liu, Xiaofeng Wang, and Rui Wang. Mrrope: Mixed-radix rotary position embedding, 2026. URL https://arxiv.org/abs/2601.22181.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Musique: Multihop questions via single-hop question composition, 2022. URL https://arxiv.org/abs/2108.00573.

Siyuan Wang, Gaokai Zhang, Li Lyna Zhang, Ning Shang, Fan Yang, Dongyao Chen, and Mao Yang. Loongrl: Reinforcement learning for advanced reasoning over long contexts, 2025. URL https://arxiv.org/abs/2510.19363.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark, 2024. URL https://arxiv.org/abs/2406.01574.

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

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering, 2018. URL https://arxiv.org/abs/1809.09600.

Hongli Yu, Tinghong Chen, Jiangtao Feng, Jiangjie Chen, Weinan Dai, Qiying Yu, Ya-Qin Zhang, Wei-Ying Ma, Jingjing Liu, Mingxuan Wang, and Hao Zhou. Memagent: Reshaping long-context llm with multi-conv rl-based memory agent, 2025. URL https://arxiv.org/abs/2507.02259.

Jingyang Yuan, Huazuo Gao, Damai Dai, Junyu Luo, Liang Zhao, Zhengyan Zhang, Zhenda Xie, Y. X. Wei, Lean Wang, Zhiping Xiao, Yuqing Wang, Chong Ruan, Ming Zhang, Wenfeng Liang, and Wangding Zeng. Native sparse attention: Hardware-aligned and natively trainable sparse attention, 2025. URL https://arxiv.org/abs/2502.11089.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models, 2023. URL https://arxiv.org/abs/2311.07911.

Appendix

A Agent Trajectory Compilation Examples

Figures 5, 6, and 7 show complete compilation examples for the Search, SWE, and SQL agents. All three follow the same ACC pipeline described in the main text.

Figure 6 shows a compiled trajectory for the SWE agent. The environment presents a partial codebase snapshot containing both files relevant to the bug and irrelevant distractors. The agent opens files selectively to locate the issue, and ACC compiles the opened file contents into a long-context background while shuffling in unopened distractors.

Figure 7 shows a compiled trajectory for the SQL agent. The environment presents a relational table that encodes a multi-hop graph structure. In the original trajectory, the agent issues SQL queries to perform recursive traversals. ACC compiles the full contents of the relevant table into a long-context background, enabling the model to perform multi-hop relational reasoning directly over the assembled records without SQL query execution.

B Extended Results on General Long-Context Tasks

Table 7 reports results on general long-context benchmarks, including multi-hop QA (HotpotQAYang et al. (2018), MuSiQueTrivedi et al. (2022)), long-document understanding (NarrativeQAKočiský et al. (2017)), and comprehensive long-context suite (LongBench-V2Bai et al. (2025)). ACC yields modest gains on these tasks.

Table 7: Extended long-context benchmark results (avg@3). Numbers in parentheses show improvement over the Qwen3-30B-A3B-Thinking baseline.

Model	LB-V2	HotpotQA	MuSiQue	NarrQA
Base Model				
Qwen3-30B-A3B-Thinking	47.87	87.00	70.10	85.00
Our Method				
Qwen3-30B-A3B-Thinking + ACC (Ours)	**48.90** (+1.03)	**88.50** (+1.50)	**70.40** (+0.30)	**85.50** (+0.50)
Strong Baseline				
Qwen3-235B-A22B-Thinking	59.76	90.00	73.30	88.50

C Data Overlap Experiment Details

Question Extraction. For each training trajectory, we parse the dialogue and keep only the user turns. We then apply a lightweight rule-based extractor to obtain the core question. Benchmark questions undergo the same cleaning pipeline to retain only the problem statement. The extracted questions are whitespace-normalized and truncated to 3,000 characters as a safety bound, though most questions are far shorter after extraction.

Embedding. We encode all cleaned questions with `all-MiniLM-L6-v2`. The model outputs 384-dimensional vectors, which we normalize to unit length. Cosine similarity is used for all distance computations.

Search Agent Trajectory Compilation Example

Original Question-Answer Pair:

Question: Which track, credited as a cover of the French new-wave group, appears as a European-bonus track on a French symphonic black-metal band's early-2000s album, is featured as a roughly five-minute live performance on that group's mid-1990s live album recorded for radio, and is listed as the third song on Disc One of the group's late-1990s live collection?
Groundtruth: Les Tzars

Original Agentic Trajectory (abridged):

```
Turn 1 (Search): "French symphonic black metal Indochine
                 cover European bonus track"
Result: [Doc A] Anorexia Nervosa - Redemption Process...
        8. Les Tzars (Indochine Cover)... limited press...
        [Doc B] Indochine is a French new wave band...
        [Doc C] Eating disorders treatment and recovery...
-> (Visit) Doc A

Turn 2 (Search): "Indochine live album 1994 radio Les Tzars"
Result: [Doc D] Radio Indochine is the second live album...
        released in 1994... Les tzars - 5:12...
        [Doc E] Indochine au Zenith 1986 live album...
        [Doc F] Neuropsychological assessment of ED...
-> (Visit) Doc D

Turn 3 (Search): "Indochine Indo Live 1997 Disc One track"
Result: [Doc G] INDO LIVE is the third live album...
        released in 1997... Disc One... Les tzars...
        [Doc H] Indochine 13 album 2017...
        [Doc I] Hiroshima mon amour 1959 film...
-> (Visit) Doc G

Turn 4 (Answer): Les Tzars ✓
```

ACC-Compiled Question-Answer Pair (abridged):

```
Question: Please reason based on the following documents and answer the question.
Which track, credited as a cover of the French new-wave group, appears as a European-bonus
track...

Documents:
[Doc G] INDO LIVE is the third live album by French new wave
band Indochine... released in 1997... Disc One... Les tzars...
[Doc C] Eating disorders treatment and recovery... (distractor)
[Doc A] Anorexia Nervosa - Redemption Process... 8. Les Tzars
(Indochine Cover)... limited press of six thousand copies...
[Doc D] Radio Indochine is the second live album by French new
wave band Indochine... released in 1994... Les tzars - 5:12...


Answer: Les Tzars
```

Figure 5: **Search Agent Trajectory Compilation Example.** The top section shows the original question and ground truth answer. The middle section shows the original agentic trajectory (documents visited are highlighted in blue, documents returned by search but never visited are highlighted in red). The bottom section shows the ACC compiled QA.

The model processes at most 256 tokens internally, so the effective input is the leading segment of each extracted question.

Dimensionality Reduction. We project the embeddings into two dimensions using UMAP with the following fixed settings: 15 nearest neighbors, minimum distance 0.3, cosine metric, PCA initialization, and random seed 42. The PCA initializer avoids the spectral initialization failure that can occur on densely connected graphs.

Metrics. All quantitative indicators are computed on the original 384-dimensional embeddings. UMAP coordinates are used only for visualization.

- **Average Nearest-Neighbor Cosine Similarity.** For each benchmark question, we identify the most similar training sample by cosine similarity and average these maxima across all benchmark instances.
- **Center Cosine Distance.** We compute the normalized mean embedding vector for the training set and for each benchmark, then take the cosine distance between these centroids (i.e., one minus their cosine similarity).
- **Linear Classifier AUC.** We train a logistic regression classifier to distinguish training samples from benchmark samples and report the area under the ROC curve.

D Attention Analysis Experiment Details

Setup. We analyze the baseline model (Qwen3-30B-A3B-Thinking) and the ACC-trained checkpoint. Both models are loaded with `AutoModelForCausalLM` in bfloat16 precision via `device_map="auto"`. To ensure the attention tensors are accessible, we force the attention implementation to eager mode, avoiding fused kernel paths that do not expose the full 4D attention matrix.

Layer and Distance Binning. We restrict the analysis to the three layers with the most significant attention changes for each task (indices 36, 30, 42 for GraphWalks and indices 15, 45, 17 for MRCR). For each head in these layers, we extract the causal attention matrix and bin token distances into 32 equal-width intervals ranging from 0 to the sequence length minus one. For each distance bin, we aggregate attention weights along the corresponding off-diagonals of the lower-triangular attention matrix and compute the per-head mean.

Metric Definition. For layer l and head h, let $A^{(l,h)} \in \mathbb{R}^{T\times T}$ denote the causal attention matrix where T is the sequence length. Let $[e_0, e_1, \ldots, e_B]$ denote the bin edges where $B = 32$. For each distance bin b spanning $[e_b, e_{b+1})$, we aggregate attention weights along the lower-triangular off-diagonals (including the main diagonal at $d = 0$):

$$\mathcal{D}_b = \{d \in \mathbb{Z} : e_b \le d < e_{b+1}\}.$$

The per-head per-bin mean is computed by averaging over all token positions that fall into those off-diagonals:

$$m_{l,h,b} = \frac{\sum_{d\in\mathcal{D}_b}\sum_{i=d}^{T-1} A^{(l,h)}_{i,i-d}}{\sum_{d\in\mathcal{D}_b}(T-d)}.$$

The per-layer per-bin mean is then obtained by averaging over all heads in the layer:

$$\mu_{l,b} = \frac{1}{H}\sum_{h=1}^{H} m_{l,h,b},$$

where H is the number of heads in layer l.

The reported heatmap shows the delta between the SFT and baseline models:

$$\Delta_{l,b} = \mu_{l,b}^{\text{SFT}} - \mu_{l,b}^{\text{Base}}.$$

Each cell in the heatmap corresponds to one layer-distance pair (l, b). Positive values indicate increased attention mass at that distance after ACC training.

For the top-head analysis, we first compute the mean attention over the tail bins (the last 25% of distance bins) for each head:

$$\tau_{l,h} = \frac{1}{|B_{\text{tail}}|} \sum_{b \in B_{\text{tail}}} m_{l,h,b},$$

where B_{tail} indexes the farthest distance bins. The per-head tail delta is:

$$\delta_{l,h}^{\text{tail}} = \tau_{l,h}^{\text{SFT}} - \tau_{l,h}^{\text{Base}}.$$

This metric is used to rank and identify heads with the strongest far-range attention change.

Statistics and Visualization. Attention statistics are aggregated across evaluation samples and averaged per head and per distance bin.

E Expert Routing Visualization Experiment Details

Models and Layers. We compare the baseline model (Qwen3-30B-A3B-Thinking) and the ACC-trained checkpoint. We restrict the analysis to three layers with the most significant expert routing changes for each task (indices 42, 40, 7 for GraphWalks and indices 17, 16, 15 for MRCR).

Dataset and Sampling. We randomly sample 32 examples from the evaluation splits of GraphWalks and MRCR, respectively. Each example is tokenized and fed through both models in inference mode to collect router statistics.

Metric Definition. For each token position t, layer l, and expert e, the router produces logits $z_{l,t} \in \mathbb{R}^E$ (E is the number of experts). Let $g_{l,t} \in \{0,1\}^E$ be the top-k gating indicator where $g_{l,t,e} = 1$ if expert e is among the top-k selected experts for token t at layer l. We define the **top-k frequency** of expert e in token group i as

$$f_{l,e}^{(i)} = \frac{1}{|S_i|} \sum_{t \in S_i} g_{l,t,e},$$

where S_i is the set of token indices belonging to group i. Let $\mathcal{L}_{\text{task}}$ denote the set of three layers with the largest mean absolute expert routing delta for the target task. The reported heatmap shows the delta

$$\Delta f_e^{(i)} = \frac{1}{|\mathcal{L}_{\text{task}}|} \sum_{l \in \mathcal{L}_{\text{task}}} \left(f_{\text{SFT},l,e}^{(i)} - f_{\text{Baseline},l,e}^{(i)} \right),$$

where the per-layer expert frequency $f_{l,e}^{(i)}$ is averaged across the selected layers $l \in \mathcal{L}_{\text{task}}$.

Token Grouping. The full sequence is divided into 32 equal-length groups via linear binning of token indices (i.e., group i covers positions $[i \cdot L/32, (i+1) \cdot L/32)$). This relative-position grouping allows comparison across variable-length sequences.

Expert Selection. We rank experts by the mean absolute delta across all groups and layers, then visualize the top 20 experts with the largest change. This avoids cluttering the figure with experts whose routing patterns remain nearly unchanged after SFT.

Implementation. To collect router logits without modifying model weights, we temporarily wrap the MoE module's forward pass in the selected three layers, extract the router logits during the forward pass, and immediately restore the original forward function. Statistics are aggregated incrementally across the 32 samples using running means to avoid storing large intermediate tensors.

F SWE TRACE JUDGE PROMPT

For SWE trajectories, retaining only rationales that reach the correct patch is not sufficient, since a correct final patch can still be paired with post hoc or hallucinated reasoning. We therefore apply an additional filtering step with a DeepSeek-V3.2-Thinking judge, and the exact prompt is shown in Figure 8. The judge receives the user prompt, the candidate reasoning trace, the predicted patch, and the ground truth patch, and it labels each trace as KEEP, WEAK KEEP, or DROP. A trace receives KEEP only when its reasoning addresses the specific issue rather than generic filler, correctly identifies the relevant files and functions, states a root cause that is consistent with the predicted patch, and contains no hallucinated APIs or files. The ground truth patch serves only as a consistency reference, and the predicted patch is not required to match it textually. Only traces labeled KEEP or WEAK KEEP are retained.

SWE Agent Trajectory Compilation

Original Question-Answer Pair:

Question: Fix the slow transpilation of `ExcitationPreserving` to basis gates `u1, u2, u3, cx`. Transpilation time grows from 0.8 s (6 qubits) to over 3988 s (24 qubits).
Groundtruth: Wrap `global_phase` additions with `sympy evaluate(False)` in `basis_translator.py` and `dagcircuit.py`.

Original Agentic Trajectory (abridged):

```
Turn 1 (Open): qiskit/transpiler/passes/basis/basis_translator.py
Content: [File A] ... dag.global_phase +=
        bound_target_dag.global_phase ... in run() ...
-> (Examine) File A
Turn 2 (Open): qiskit/dagcircuit/dagcircuit.py
Content: [File B] ... self.global_phase +=
        in_dag.global_phase ... in substitute_node_with_dag() ...
-> (Examine) File B
Turn 3 (Open): qiskit/transpiler/passes/basis/basis_translator.py
Content: [File A] ... dag.global_phase +=
        bound_target_dag.global_phase ...
-> (Modify) File A
        from sympy import evaluate
        with evaluate(False):
            dag.global_phase += bound_target_dag.global_phase
Turn 4 (Open): qiskit/dagcircuit/dagcircuit.py
Content: [File B] ... self.global_phase +=
        in_dag.global_phase ...
-> (Modify) File B
        from sympy import evaluate
        with evaluate(False):
            self.global_phase += in_dag.global_phase
Turn 5 (Submit): Tests passed ✓
```

ACC-Compiled Question-Answer Pair (abridged):

```
Question: Fix the slow transpilation of ExcitationPreserving to
basis gates u1, u2, u3, cx...

Context:
[File B] qiskit/dagcircuit/dagcircuit.py ...
self.global_phase += in_dag.global_phase ... in
substitute_node_with_dag() ...
[File A] qiskit/transpiler/passes/basis/basis_translator.py ...
dag.global_phase += bound_target_dag.global_phase ... in run() ...
[File C] qiskit/pulse/instructions/phase.py ...
(distractor:  present in codebase snapshot but never opened)

Answer:
<patch>
--- a/qiskit/dagcircuit/dagcircuit.py
+++ b/qiskit/dagcircuit/dagcircuit.py
@@ -990,7 +990,9 @@
         if in_dag.global_phase:
-            self.global_phase += in_dag.global_phase
+            from sympy import evaluate
+            with evaluate(False):
+                self.global_phase += in_dag.global_phase
... (analogous change in basis_translator.py)
</patch>
```

Figure 6: **SWE Agent Trajectory Compilation Example.** The agent opens files from the codebase snapshot and decides either to (Examine) or to (Modify) them to fix the bug. ACC retains only the *opened* files and shuffles in one unopened distractor (highlighted in red) to form the compiled QA.

SQL Agent Trajectory Compilation

Original Question-Answer Pair:

Question: User `'u_337bc99c'` (Region: East) has applied for Diamond Membership. Verify their complete referral lineage by walking upward through the referral graph following active edges (`status='active'`) with `edge_level >= 3` within region `'East'`. Prioritize higher `commission_rate` at each step; break ties by alphabetical `referrer_id`. Report the final root referrer.

Groundtruth: `u_ea8952bc`

Original Agentic Trajectory (abridged):

```
Turn 1 (Execute SQL):
WITH RECURSIVE upstream AS (
  SELECT referrer_id, referred_id, commission_rate,
         edge_level, status, region, 1 AS depth
  FROM referrals
  WHERE referred_id = 'u_337bc99c' AND status = 'active'
    AND edge_level >= 3 AND region = 'East'
  UNION ALL
  SELECT r.referrer_id, r.referred_id, r.commission_rate,
         r.edge_level, r.status, r.region, u.depth + 1
  FROM upstream u
  JOIN referrals r ON r.referred_id = u.referrer_id
  WHERE r.status = 'active' AND r.edge_level >= 3
    AND r.region = 'East'
)
SELECT referrer_id FROM upstream
ORDER BY depth DESC LIMIT 1;

Result: u_ea8952bc

Turn 2 (Answer): u_ea8952bc ✓
```

ACC-Compiled Question-Answer Pair (abridged):

```
Question: User 'u_337bc99c' (Region: East) has applied for Diamond
Membership. Walk upward through the referral graph following active
edges with edge_level >= 3 within region 'East', prioritizing higher
commission_rate. Report the final root referrer.

Context:
[Table: referrals]
| ... | referrer_id | level | rate | ltv    | region |
| ... | u_4df289c8  | 5     | 0.15 | 82.19  | East   |
| ... | u_fb9c918a  | 3     | 0.08 | 201.42 | East   |
| ... | u_be13d04d  | 4     | 0.12 | 336.31 | East   |
| ... | u_a4663191  | 5     | 0.15 | 264.93 | East   |
| ... | u_e1ce60aa  | 3     | 0.05 | 443.63 | East   |
| ... | u_54ef1623  | 3     | 0.05 | 355.49 | East   |
| ... | u_ea8952bc  | 5     | 0.10 | 108.68 | East   |
| ... | u_b7c33fc2  | 3     | 0.10 | 264.93 | East   |
| ... | u_63490c55  | 3     | 0.05 | 201.42 | North  |
| ... | u_d28bb4a9  | 1     | 0.12 | 254.94 | East   |
| ... | u_8eda5bd1  | 3     | 0.08 | 254.94 | Central|
... (remaining rows omitted)

Answer: u_ea8952bc
```

Figure 7: **SQL Agent Trajectory Compilation Example.** The top section shows the original question and ground truth answer. The middle section shows the original agentic trajectory, where the agent executes a recursive SQL query to traverse the referral graph. The bottom section shows the ACC-compiled QA, where the complete contents of the relevant database table are assembled into the provided long-context background. The ellipsis column (`...`) indicates additional fields that are present in the compiled context but omitted here for brevity.

SWE Reasoning Trajectory Judge Prompt

```
You are judging the quality of a SWE-bench reasoning trajectory for training data.
The final patch has already passed a separate correctness judge. Your task is to judge
whether the reasoning trajectory itself is credible enough to use as supervision.
User prompt
{user_prompt}
Reasoning trajectory inside think tags
{cot}
Predicted final patch
{predicted_patch}
Reference ground-truth patch
{ground_truth}
This is a no-distractor example. The prompt is expected to contain mostly issue-relevant code
context,
not a large set of unrelated distractor files.

Judge KEEP only if all are true
1. The reasoning is specific to the given issue and code context, not generic filler.
2. It identifies the relevant files/functions and explains why the provided context is sufficient.
3. The stated root cause is consistent with the predicted final patch.
4. The predicted final patch follows naturally from the reasoning.
5. The reasoning does not contain major hallucinated APIs/files or contradictions.

Judge WEAK_KEEP if the patch appears supported but the trajectory is partially generic,
incomplete,
or only loosely explains the implementation. WEAK_KEEP is acceptable as lower-confidence training
data,
but it should not be counted as the high-confidence trajectory subset.

Judge DROP if the reasoning is mostly post-hoc handwaving, contradicts the predicted patch, cites
nonexistent context,
skips or misidentifies the key root cause, or explains a materially different solution than the
predicted patch.

Use the ground-truth patch only as a reference for consistency. Do not require the predicted
patch to be textually
identical to the ground truth if the reasoning and predicted patch are semantically coherent.

Output exactly one of
<judgment>KEEP</judgment>
<judgment>WEAK_KEEP</judgment>
<judgment>DROP</judgment>
<reason>one concise sentence</reason>
```

Figure 8: **Prompt for the DeepSeek-V3.2-Thinking judge** used to filter SWE reasoning traces. The judge evaluates specificity, localization, root-cause consistency, and hallucination before assigning a KEEP, WEAK_KEEP, or DROP label.