RECURSIVE LANGUAGE MODELS MEET UNCERTAINTY: THE SURPRISING EFFECTIVENESS OF SELF-REFLECTIVE PROGRAM SEARCH FOR LONG CONTEXT

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-context handling remains a core challenge for language models: even with extended context windows, models often fail to reliably extract, reason over, and use the information across long contexts. Recent works like Recursive Language Models (RLMs) have approached this challenge by agentic way of decomposing long contexts into recursive sub-queries through programmatic interaction at inference. While promising, the success of RLMs critically depends on how these trajectories of context-interaction programs are selected, which has remained unexplored. In this paper, we study this problem and introduce Self-Reflective Program Search for Long Context (SRLM), a framework that augments programming-based context interaction with uncertainty-aware self-reflection. SRLM leverages three intrinsic signals: self consistency, reasoning trace length, and verbalized confidence. These serve as complementary indicators of a model's internal uncertainty, and the model uses them to evaluate and compare candidate context-interaction programs. Extensive experiments across diverse benchmark datasets, context lengths, and backbone models, show that SRLM consistently outperforms state-of-the-art baselines, yielding up to 22% improvement over RLMs under the same time budget. Our findings show that recursion itself is not the primary driver of performance in RLMs, and a simple self-reflective program search can match or surpass RLM without requiring self-query or explicit recursion mechanisms. We find that for context lengths within the model's context window, RLMs with recursion often degrade performance relative to the base model, whereas SRLM yields consistent and robust gains across both short and long contexts. We also find that RLM is less effective in tasks with semantically intensive nature, where heuristic program search is insufficient and broader contextual understanding is required, while self-reflection in SRLM provides a semantic signal that better steers reasoning in these challenging long-context scenarios.

1 INTRODUCTION

Large language models are increasingly deployed in settings where long-context understanding is not optional but unavoidable. Modern applications from deep research agents (Huang et al., 2025) and web browsing systems (Chen et al., 2025) to coding assistants (Jimenez et al., 2024) and self-improving agents (Zhang et al., 2025d) routinely demand reasoning over hundreds of thousands to millions of tokens spanning documents, logs, repositories, and interaction histories. Despite rapid progress in extending models' context windows, effective utilization of long contexts still remains challenging. Empirical studies show that even in frontier models with very large context windows, performance degrades with context length in ways that are well-documented but not yet solved: models lose track of salient details, fail to reliably extract, integrate, and reason over relevant information across distant positions, and are easily distracted by irrelevant content (Liu et al., 2023; Hong et al., 2025; Du et al., 2025).

The research community has approached this challenge from several angles. One direction has been to target this problem at the model level for example through architecture sparsity mechanisms (Tang

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.

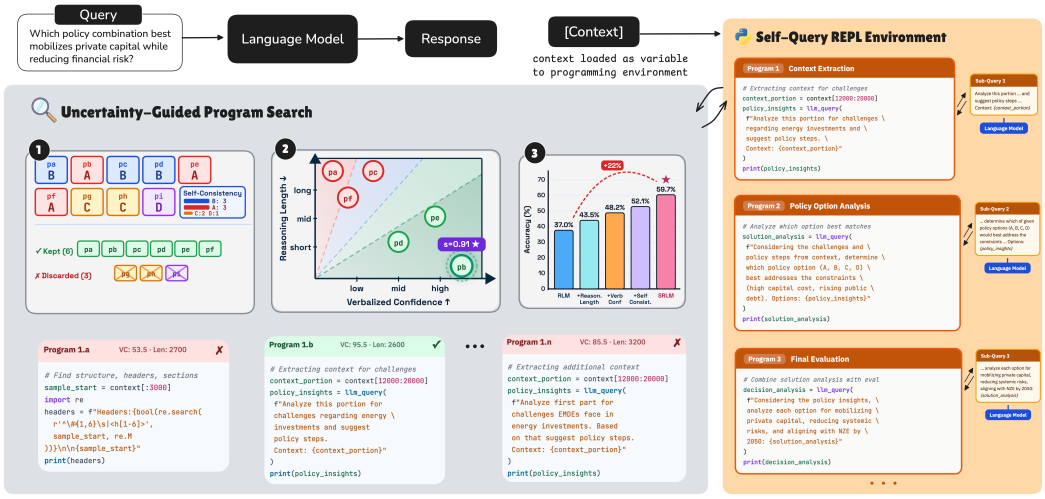


Figure 1: **Overview of SRLM**, a framework that augments programmatic context interaction reasoning with uncertainty-aware self-reflection. The language model operates in a self-query execution programming environment where the context is externalized as a variable, and generates programs that query and interact with context. Meanwhile, three complementary uncertainty signals (self-consistency, reasoning trace length, and verbalized confidence) are used to guide self-reflective programming trajectory selection without external supervision, enabling more robust and semantically grounded long-context reasoning.

et al., 2024; Gao et al., 2024; Lai et al., 2025), state-space models (Gu & Dao, 2023; Dao & Gu, 2024; Waleffe et al., 2024), retrieval-based hybrid models (Jin et al., 2024; Wang et al., 2023), or KV cache compression (Eyuboglu et al., 2025), reducing the effective cost of processing long sequences. Another direction has been at the data and training level where models are specifically trained on longer sequences or curating corpora that reward reasoning over long-horizons (Fu et al., 2024; Zhao et al., 2024). A more recent promising direction treats long-context reasoning as a search problem at inference-time, leaving the model unchanged and instead restructuring how it interacts with context (Wu et al., 2025; Zhang et al., 2025d). Chunking and summarization pipelines break long contexts into manageable pieces; retrieval systems surface relevant passages on demand; and agent-style frameworks issue iterative queries over the context, building up answers through a sequence of focused interactions.

Recursive Language Models (RLMs) (Zhang et al., 2025a) represent the current state of the art in this inference-time context handling paradigm. Instead of processing long context with millions of tokens directly with model, RLM treats the context as an external variable within a programming environment, and allows the model to generate programs that query, slice, and recursively interact with the context. By externalizing context interaction with program execution, RLM has shown to extend the model's effective reasoning horizon beyond what prompting typically allows.

However, this framing introduces a largely unexplored dimension of the problem. The quality of long-context reasoning in RLM is governed not only by the model's capacity to process extended context, but also by the mechanism used to select trajectories of context-interaction programs. At each step, the model must decide which context segment to inspect, how to formulate intermediate self-queries, what sub-questions to pose, and how to aggregate these programming steps and partial results. The final prediction is therefore highly sensitive to the specific program trajectory instantiated during context interaction reasoning. Despite this, RLM currently predominantly rely on fixed recursion schemes, lacking a principled mechanism for evaluating and selecting among alternative reasoning trajectories. This raises a central question: *Is recursion itself the key ingredient for long-context reasoning, or is the real bottleneck how we select among candidate interaction programs under uncertainty?*

In this work, we investigate this question and introduce Self-Reflective Program Search for Long Context Method (**SRLM**), a framework that augments programming-based context interaction with uncertainty-aware self-reflection (Figure 1). SRLM leverages three complementary signals

(self-consistency, reasoning length, and verbalized confidence) as proxies for the model's internal uncertainty, enabling principled comparison of context-interaction trajectories through model's self-reflection without requiring external supervision. Through extensive comparison experiments across diverse benchmarks, varying context lengths, and multiple backbone models, we observe that SRLM consistently outperforms state-of-the-art baselines, yielding up to 22% improvement over RLM under the same wall-clock time budget.

Beyond empirical improvements, our analysis provides several insights into programming-based context-interaction frameworks like RLM and its key components. First, we find that recursion is not the primary driver of performance. A simple self-reflective program search can match or even surpass RLM without relying on explicit recursion or self-query mechanisms under the same latency. Second, the recursive self-query procedure is often more sensitive to context-length variations than self-reflection. In particular, when the context length falls within the model's native context window, recursive RLM reasoning can degrade performance relative to the base model, whereas SRLM yields more robust and consistent improvements across both short and long contexts. Finally, we observe that RLM is less effective on semantically intensive tasks where heuristic program search is insufficient. In such settings, the simple self-reflection mechanism in SRLM provides a higher-level semantic signal that more effectively steers reasoning. Together, these findings reposition recursion as one component of long-context reasoning rather than its defining feature, and suggest that uncertainty-aware self-reflection may serve as a simple yet effective alternative for building robust context-interaction frameworks.

More broadly, our goal in this paper is not just introducing a novel method but to better understand the new programming-based context-interaction frameworks and the role of their core components. Our study highlights the critical importance of programming trajectory selection in long-context interaction and suggests that improving how models explore and evaluate candidate interaction programs may be as important as subagent delegation and extending context length itself. We hope that these findings help guide the development of richer and more reliable long-context reasoning frameworks in future work.

2 METHODOLOGY

2.1 PROBLEM FORMULATION

Let q denote a natural language query and $\mathcal{C} = (c_1, c_2, \ldots, c_N)$ a long context of N tokens, where $N \gg L$ with L being the model's effective context window. Rather than feeding $\mathcal{C}$ directly to model, we follow Zhang et al. (2025a) and treat context as an *external variable* accessible within a sandboxed execution programming environment. A context-interaction program $p = (p_1, p_2, \ldots, p_T)$ is a sequence of T executable operations, e.g., slicing, querying, or aggregating over $\mathcal{C}$, each generated autoregressively and executed in the REPL, producing an intermediate execution state: $e_t = \textsc{Exec}(p_t,\ e_{t-1},\ \mathcal{C})$, where $e_0 = \varnothing$. The terminal step yields the program output $\text{out}(p) \in \mathcal{A}$ over answer space $\mathcal{A}$. A key distinction from Zhang et al. (2025a) is that SRLM does *not* require programs to instantiate explicit self-query sub-calls or recursive model invocations as tool calls. This decouples quality of context interaction from the structure of recursion, and shifts the focus of long-context reasoning improvement to the selection mechanism over candidate context-interaction program trajectories.

2.2 SRLM: SELF-REFLECTIVE PROGRAM SEARCH FOR LONG CONTEXT

Given query q and context $\mathcal{C}$, K candidate programs are independently selected from the model policy π_θ: $p^{(k)} \sim \pi_\theta(\cdot \mid q, \mathcal{C}), \quad k = 1, \ldots, K$. Each $p^{(k)}$ constitutes a distinct reasoning trajectory over $\mathcal{C}$, differing in which context segments are inspected, how sub-problems are decomposed, and the confidence with which intermediate conclusions are drawn. We propose a self-reflective program search approach for long-context reasoning that draws on three complementary uncertainty signals: *sampling-based uncertainty (self-consistency)*, *semantic uncertainty (verbalized confidence)*, and *behavioral uncertainty (reasoning trace length)*. Notably, all these three signals are derived from the model's own generation process, requiring no verifier, reward model, or external labeled data.

2.2.1 UNCERTAINTY SIGNALS

Sampling-based Uncertainty (Self-Consistency). As per Tao et al. (2025), a natural first-order uncertainty quantification arises directly from the sampling distribution over programs. Given K independent draws from π_θ, the empirical frequency of any candidate answer $a \in \mathcal{A}$ serves as an estimate of the model's marginal confidence in that answer, i.e., $\mathrm{prob}(a) = \frac{1}{K}\sum_{k=1}^{K} \mathbf{1}\big[\mathrm{out}(p^{(k)}) = a\big] \approx \mathbb{P}_{\pi_\theta}\big[\mathrm{out}(p) = a \mid q, \mathcal{C}\big]$. The plurality answers $\hat{a} = \arg\max_{a\in\mathcal{A}} \mathrm{prob}(a)$ maximize this empirical confidence, and we retain the consistent candidate set as the subset of programs that agree with $\hat{a}$: $\mathcal{S} = \big\{p^{(k)} \in \mathcal{P} : \mathrm{out}(p^{(k)}) = \hat{a}\big\} \subseteq \mathcal{P}$. This step performs implicit verification through self-consistency (Wang et al., 2022), however, self-consistency is a coarse uncertainty signal that operates only at the level of final outputs and is insensitive to the quality of the trajectory that produced them. Programs in $\mathcal{S}$ may share the same answer $\hat{a}$, yet may differ substantially in how they arrived at it: which context segments they inspected, how confidently they resolved each sub-problem, and how much deliberation they required. Selecting reliably among these candidates demands finer-grained uncertainty measures.

Semantic Uncertainty (Verbalized Confidence). Inspired by Xiong et al. (2023), to obtain a step-level semantic uncertainty signal, we elicit the model's own assessment of its confidence at each intermediate generation step t. Specifically, we append a structured instruction to the model's prompt, requiring it to report a confidence score for each step in a standardized format `{"confidence": `$\nu_t^{(k)}$`}`, $\nu_t^{(k)} \in (0, 100]$, where the model is instructed to be precise and nuanced in its self-assessment. This elicitation yields a per-step confidence $\nu_t^{(k)}$ reflecting the model's self-assessed certainty over its intermediate conclusion at step t (Xiong et al., 2023). Normalizing to the unit interval and aggregating in log-space over the full trace, we define the verbalized confidence score of program $p^{(k)}$ as $\mathrm{VC}(p^{(k)}) = \sum_{t=1}^{T^{(k)}} \log\Big(\nu_t^{(k)}/100\Big) \le 0$, where non-positivity follows from $\nu_t^{(k)}/100 \in (0, 1]$, and values closer to zero indicate globally higher confidence across the trajectory. Unlike self-consistency, $\mathrm{VC}(p^{(k)})$ is a semantic uncertainty measure that captures how the model endorses each intermediate reasoning step as it progressively builds toward the final answer. For more details of prompt used for this, check Appendix C.1.

Behavioral Uncertainty (Reasoning Length). While verbalized confidence relies on the model's explicit self-report at each step, we additionally exploit an implicit behavioral signal as the total token length of the generated trace. Let $\ell_t^{(k)}$ denote the number of reasoning and output tokens at step t; we define $\mathrm{Len}(p^{(k)}) = \sum_{t=1}^{T^{(k)}} \ell_t^{(k)}$. We interpret this quantity as a proxy for epistemic effort. Intuitively, when a model is uncertain, it tends to generate longer, more deliberative traces, whereas confident and well-grounded reasoning is often associated with more concise outputs (Devic et al., 2025; Shojaee et al., 2025). Importantly, trace length provides a signal complementary to verbalized confidence (Devic et al., 2025). Unlike self-reported confidence scores, it requires no explicit elicitation and is derived solely from observable generation statistics. As such, it offers an alternative fine-grained window into internal uncertainty that is not directly subject to miscalibration in the model's stated confidence.

2.2.2 JOINT UNCERTAINTY-GUIDED SELECTION

The three uncertainty signals (self-consistency, verbalized confidence, and trace length) are complementary proxies of model uncertainty, each capturing a distinct aspect of the model's internal state. As our empirical results demonstrate (Section 3.8), combining these signals yields a richer uncertainty characterization that more effectively guides program search over long-context interaction programs than any individual signal alone. Within the consistent candidate set $\mathcal{S}$ (where self-consistency has already been enforced), we unify the remaining two signals into a joint uncertainty score of $s(p) = \mathrm{VC}(p) \cdot \mathrm{Len}(p)$ where lower values of $s(p)$ indicate better candidates. By construction, $s(p) \le 0$ since $\mathrm{VC}(p) \le 0$ and $\mathrm{Len}(p) > 0$. Intuitively, this score penalizes programs that express low confidence or require excessively long reasoning traces—both indicators of uncertainty. The optimal program is then selected as $p^* = \arg\max_{p\in\mathcal{S}} s(p)$, with final prediction $\hat{y} = \mathrm{out}(p^*)$. Together, these three uncertainty signals form a coherent, self-reflective framework that effectively guides program search in SRLM without requiring any external supervision.

Table 1: Performance comparison of SRLM against baselines on long-context benchmarks from Zhang et al. (2025a). Results report accuracy (%) on LongBench-v2 CodeQA, BrowseComp+ (1K documents), and OOLONG (131K tokens). SRLM consistently outperforms all baselines, achieving up to 22% improvement over RLM. * indicates context overflow; † indicates our replication of results; and **bold** shows best result per LLM backbone.

Model	LongBench-v2 (CodeQA)	BrowseComp+(1K)	OOLONG
Task Length N (tokens)	23K-4.2M	6M-11M	131K
Qwen3-Coder-480B			
Base Model	20.0*	0.0*	36.0
CodeAct (+ BM25)	24.0*	12.7	38.0
CodeAct (+ sub-calls)	26.0*	0.0	32.0
Summary agent	50.0	38.0	44.1
RLM	59.8†	37.1†	45.7†
RLM (no sub-calls)	53.8†	36.3†	39.1†
SRLM	**64.9** (↑5.1)	**59.7** (↑22.6)	**51.8** (↑6.1)
SRLM (no sub-calls)	59.0 (↑5.2)	50.1 (↑13.8)	45.9 (↑6.8)
GPT-5			
Base Model	24.0*	0.0*	44.0
CodeAct (+ BM25)	22.0*	51.0	38.0
CodeAct (+ sub-calls)	24.0*	0.0*	40.0
Summary agent	58.0	70.5	46.0
RLM	59.5†	86.0†	53.0†
RLM (no sub-calls)	65.2†	89.7†	50.5†
SRLM	68.9 (↑9.4)	92.4 (↑6.4)	**65.5** (↑12.5)
SRLM (no sub-calls)	**74.1** (↑8.9)	**94.6** (↑4.9)	60.7 (↑10.2)

3 Experiments

3.1 Datasets

Following Zhang et al. (2025a), we evaluate SRLM on three benchmarks spanning diverse long-context reasoning tasks. **BrowseComp+ (1K)** (Chen et al., 2025) is a multi-hop QA benchmark for DeepResearch (OpenAI, 2025) over a verified offline corpus of 1,000 documents, where each question requires piecing together evidence across multiple documents. Following (Zhang et al., 2025a; Sun et al., 2025), we evaluate on 150 randomly sampled instances and report accuracy. **OOLONG (131K)** (Bertsch et al., 2025) requires transformation and aggregation of input chunks, scaling linearly in processing complexity with context length. We focus on the `trec_coarse` split from OOLONG synthetic benchmark with context length 131K (50 tasks), and report scores following the original paper. **LongBench-v2 CodeQA** (Bai et al., 2024a) is a multiple-choice code repository understanding benchmark requiring reasoning over long-context of files in a codebase (50 tasks).

Beyond this, we conduct extended evaluations targeting the core research questions of this study. To characterize how context length affects SRLM and RLM, we evaluate on the **full OOLONG synthetic benchmark** (`trec_coarse` split) across context lengths from 1K to 4M tokens (≈650 tasks, 50 per length). To investigate the effect of task semantics and extend evaluation to tasks that by nature require more semantic understanding rather than heuristic search over context, we also evaluate on the **full LongBench-v2** benchmark across all domain categories beyond just CodeQA (≈500 tasks), including domains like single document QA, multi-document QA, long in-context learning, etc. For more details on statistics, context length distributions, and category breakdowns of these datasets, check Appendix B.

3.2 Baselines

We compare against a comprehensive set of task-agnostic inference-time baselines following Zhang et al. (2025a). **Base LLM** processes the full context in prompt without any programmatic inference scaffolding. **CodeAct (+BM25)** (Wang et al., 2024) is a code-executing ReAct (Yao et al., 2022) agent that receives the full context directly and is additionally equipped with a BM25 retriever (Robertson & Zaragoza, 2009) for context search as per Zhang et al. (2025a); Jimenez et al. (2023); Chen et al. (2025). **CodeAct (+sub-calls)** ablates the effect of context offloading as a variable in REPL by augmenting the CodeAct baseline with the ability to invoke sub-calls from the language model.

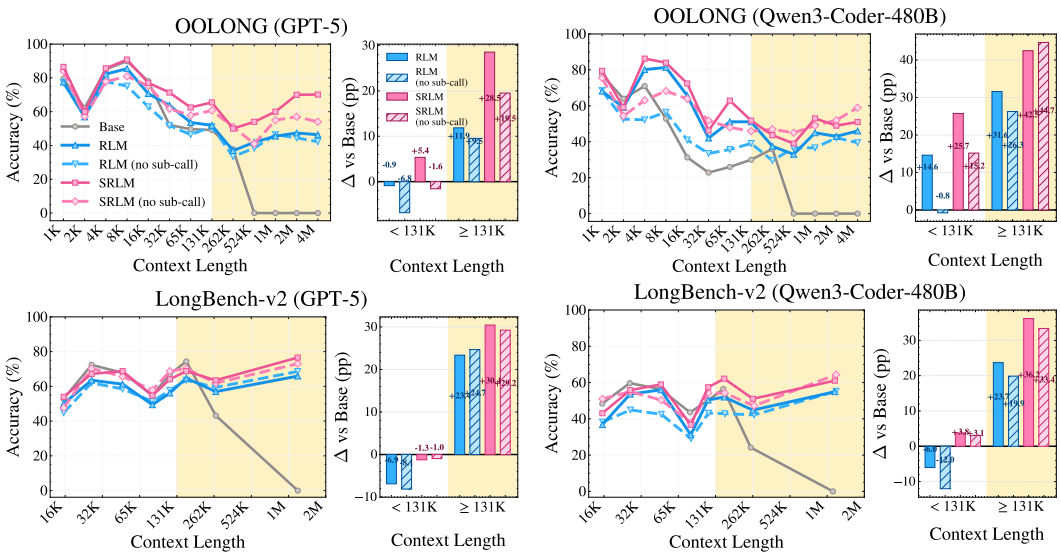


Figure 2: Performance across context lengths on OOLONG and LongBench-v2 full datasets: Line plots show accuracy of SRLM, RLM, and the base LLM across context from thousands to millions of tokens using GPT-5 **(left)** and Qwen3-Coder-480B **(right)** backbones. Bar plots show the average performance gain over the base model, separated into contexts within ($<$131K) and near/beyond ($\geq$131K) the native context window.

Summary agent also follows Sun et al. (2025); Wu et al. (2025); Yu et al. (2025) and iteratively compacts and summarizes context as the model window fills, chunking documents that exceed the context limit. **RLM** (Zhang et al., 2025a) is the current state-of-the-art approach, externalizing context as a variable in a REPL environment and issuing recursive self-queries; we consider both the recursive variant (depth one) and the **no sub-calls** variant that disables this self-query procedure. For each comparison across baseline methods, we use the same backbone models and sampling parameters.

3.3 Experimental Setup

In our experiments, we use two backbone LLMs: the open-weight Qwen3-Coder-480B-A35B (Team, 2025) and GPT-5 (Singh et al., 2025) with medium reasoning effort, with GPT-5-mini as the sub-model for the recursive calls (as per Zhang et al. (2025a)). SRLM operates in the same REPL environment as RLM and uses $K{=}8$ candidate trajectories for uncertainty-guided program search, with uncertainty signals defined in Section 4. To ensure fair wall-clock time comparison across methods, we impose execution time limits of 600 seconds per each step of trajectory for all runs. We set a maximum of 30 program interaction steps and a maximum generation length of 260K tokens for Qwen3-Coder-480B, with default API parameters for GPT-5 and GPT-5-mini calls. For verbalized confidence elicitation, we augment the original RLM prompt (As in Zhang et al. (2025a)) with a suffix requesting the self-report of internal confidence in a structured format, without modifying any other part of the prompt or reasoning procedure (see Appendix C.1 for details). For final answer evaluation, we also use GPT-5-mini as a judge across all datasets to robustly assess the correctness (check Appendix C.2 for details).

3.4 Main Results

Table 1 compares SRLM with RLM and other baselines on the long-context benchmarks from Zhang et al. (2025a). Across all datasets and both backbone LLMs, SRLM consistently performs the best, improving over the previous state-of-the-art RLM by up to 22%. Looking more closely, we find that the effect of recursion is inconsistent across backbone models. For example, with Qwen3-Coder-480B, recursion helps both RLM and SRLM, suggesting that decomposing the context into smaller sub-problems through model's self-query procedure as tool call can support long-context handling in Qwen backbone. However, under GPT-5, recursion hurts performance and the variants

without sub-calls outperform their recursive counterparts in most cases. This might indicate that when the backbone model is already strong at long-context reasoning, explicitly using self-query tool calls for context-interaction may be unnecessary or even disruptive. In contrast, the self-reflection mechanism in SRLM provides stable improvements across both of these backbones. Even without any sub-calls, SRLM often outperforms recursive RLM. This suggests that guiding through model internals and self-reflection may matter more than guiding through explicit recursive tool calls for the scope of long context. We analyze these behaviors in greater detail in Section 3.5–3.7.

3.5 Robustness Across Context Lengths

Next, we investigate how context length affects the behavior of each method. To this end, we run additional experiments on the full LongBench-v2 and OOLONG datasets, covering contexts from thousand to millions of tokens. Figure 2 compares SRLM, RLM, and the base LLM across context lengths. We also demonstrate the performance gap relative to the base model (Δ vs. base), separating results for shorter contexts well within the model's context window ($<$131K), and longer contexts near or beyond the context limit ($\geq$131K). From these results, we observe several interesting patterns. First, the advantage of SRLM becomes more pronounced as context length increases. On longer contexts, SRLM consistently provides better gains over the base model than RLM. Second, RLM is noticeably more sensitive to context length. On shorter contexts (for example <131K), RLM often underperforms the base model, indicating that recursive decomposition may not be effective on all contexts and can introduce unnecessary overhead when the context is already manageable. In contrast, SRLM remains robust and provides more consistent gains over the base model on both short and long contexts. For more detailed results over backbones and tasks, check Appendix D.1.

3.6 Is Recursion the Primary Driver of Performance in RLMs?

One of the key motivations and a central question to this study is whether recursion is the main source of gains in recursive language models (RLMs), particularly in long-context settings. Understanding this can help toward guiding the design of more effective frameworks. Conceptually, recursion can be viewed as a form of inference-time scaling through model as a tool use, i.e., the model decomposes the problem into sub-queries and recursively calls itself as a tool to interact with different parts of the context. In contrast, our self-reflective variant (SRLM without sub-calls) performs inference-time scaling through model's internals. Instead of explicitly issuing recursive tool sub-calls, it relies on implicit uncertainty-guided self-reflection to revise and refine its context-interaction programs. To study the above research question, we focus on the direct comparison of context-interaction recursive programming (RLM with sub-calls) and self-reflective programming (SRLM without sub-calls) on long-context settings ($\geq$131K tokens) across various datasets and backbones.

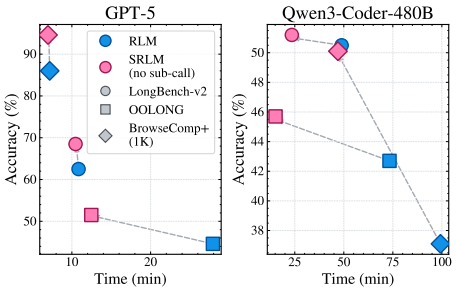


Figure 3: Accuracy versus cost pareto comparison of RLM and SRLM (no sub-call) on long-context settings under GPT-5 (**left**) and Qwen3-Coder-480B (**right**).

As shown in Figure 3, self-reflection can actually outperform recursion in both performance and cost (wall-clock time) under long-context settings. This suggests that recursion may not be the best strategy for inference-time scaling in long-context interactions, as explicit self-querying and sub-calls introduce additional overhead with only marginal performance gains. This trend is also reflected in Table 1 results. In cases where recursion helps, most of the gains over the base model come from the programmatic context-interaction procedure rather than recursion. For example, on LongBench CodeQA with Qwen3-Coder-480B, performance improves from 20 to 53.8 with RLM without sub-calls, and only further to 59.8 with recursive sub-calls which can be obtained with SRLM without sub-calls as an alternative inference time scaling method, indicating that recursion contributes only marginal gains in these long-context frameworks.

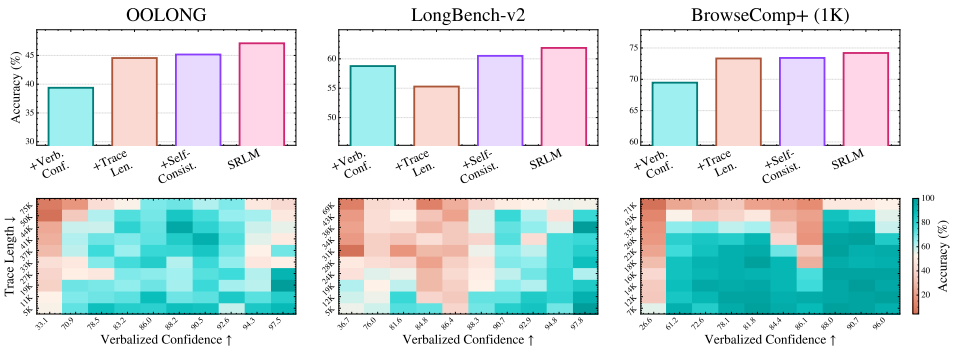


Figure 5: Results of ablation experiments across SRLM's variants (averaged across backbones and recursive/nonrecursive runs). **Top:** Contribution of each uncertainty signal and their combination in SRLM. **Bottom:** Complementary effects of semantic and behavioral uncertainty as fine-grained signals guiding self-reflection in SRLM.

3.7 Task Semantics and Limits of Recursion

Beyond overall long-context performance, it is important to examine how recursion and self-reflection behave across tasks of different natures. The tasks studied in original work of RLM (including OOLONG, BrowseComp+, and LongBench-v2 Code Repository QA) are largely search-oriented by nature. For example, in LongBench-v2, experiments focus on the Code QA category, where answering a question typically requires locating specific information across structured repository files. These contexts are modular and well-organized, making them naturally suitable for recursive and programmatic traversal. However, LongBench-v2 also includes a broader set of domains beyond code QA, such as document QA, dialogue history QA, and in-context learning. Many of these tasks require integrating evidence distributed throughout the context rather than locating specific pieces of information. To better understand robustness across diverse long-context tasks, we extend our evaluation to these additional domains.

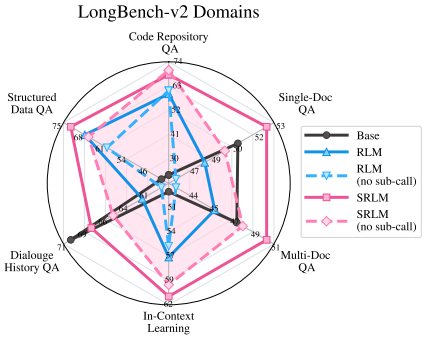


Figure 4: Comparison of SRLM, RLM and Base LLM across domains of LongBench-v2 (averaged across backbone models).

Figure 4 shows performance by domain on LongBench-v2 (averaged across backbones). We observe that the impact of recursion varies with task type: it is more effective for structured, search-oriented tasks such as Code QA and Structured Data QA, but less beneficial for tasks that are dense in context and semantically more demanding like Dialogue History QA and Document QA. In contrast, self-reflection in SRLM provides more consistent gains across these task categories.

3.8 Ablation Study

In this section, we perform ablations on the self-reflection mechanism of SRLM to understand the contribution of its components. Figure 5 reports results on OOLONG, LongBench-v2, and BrowseComp+ (1K) under long-context settings ($\geq$131K tokens), aggregated across model backbones and runs with and without recursive sub-calls. SRLM leverages three complementary uncertainty signals during self-reflection: (1) sampling-based self-consistency, (2) semantic uncertainty captured through verbalized confidence, and (3) behavioral uncertainty measured by the length of the reasoning trace. The top row of Figure 5 shows the impact of each component. As it can be observed, the full SRLM configuration consistently outperforms variants using individual signals, indicating that these uncertainty sources provide complementary benefits. The bottom row of Figure 5 examines the relation between fine-grained uncertainty metrics (verbalized confidence and reasoning trace length) and accuracy under bins with same samples. As it can be observed, performance depends jointly on

both signals, and their relation with accuracy is not strictly linear. In other words, high confidence or short traces alone do not reliably indicate correctness, however, their combination seem to provide a stronger self-reflection signal in the scope of long-context.

4 Related Works

Context Window Expansion. Prior work on long-context language modeling has focused on extending context windows through architectural and training innovations, including improved positional encoding and attention scaling (Chen et al., 2023; Ding et al., 2024), as well as sparsity mechanisms (Tang et al., 2024; Gao et al., 2024), state-space models (Gu & Dao, 2023), retrieval-augmented fine-tuning (Jin et al., 2024; Wang et al., 2023), and KV-cache compression (Eyuboglu et al., 2025). While these approaches improve scalability, empirical evidence shows performance remains limited by an effective context length.

Agentic Long-Context Approaches. An alternative line of research treats long-context processing as an agentic problem, using LLMs to iteratively interact with context via programmatic strategies. Recursive Language Models (RLMs) (Zhang et al., 2025a) decompose and recursively process context through program execution, outperforming monolithic prompting and summarization baselines. Related approaches include code-execution agents such as CodeAct (Wang et al., 2024), summarization-based agents such as ReSum (Wu et al., 2025), and memory-augmented agents such as Mem0 (Chhikara et al., 2025) and G-Memory (Zhang et al., 2025c). However, these methods typically rely on surface-level heuristics rather than semantic understanding, limiting their effectiveness on semantically dense tasks requiring deep comprehension over long-context.

Confidence Estimation. Several approaches extract uncertainty signals from LLMs without fine-tuning. Sampling-based methods estimate confidence via agreement across multiple generations (Wang et al., 2022; Kuhn et al., 2023; Manakul et al., 2023), while verbalized confidence prompts models to report calibrated scores directly (Xiong et al., 2023; Yoon et al., 2025)—often outperforming token probability baselines (Tao et al., 2025). Other approaches use token likelihoods (Duan et al., 2024) or learned probes (Zhang et al., 2025b). Recent work further identifies reasoning trace length as an implicit uncertainty proxy, noting that incorrect trajectories tend to be longer and more deliberative (Marjanović et al., 2025; Ballon et al., 2025; Devic et al., 2025; Vanhoyweghen et al., 2025). Our work combines sampling-based consistency, verbalized confidence, and reasoning length as complementary signals to guide uncertainty-aware self-reflection during program search over long contexts.

5 Conclusion

In this paper, we study long-context reasoning through context-interaction programming and introduce SRLM, a self-reflective program search framework that leverages intrinsic uncertainty signals—self-consistency, reasoning trace length, and verbalized confidence—to guide context interaction. Across diverse benchmarks, context lengths, and backbone models, SRLM consistently improves performance, achieving gains of up to 22% over RLM in long-context settings. Our analysis suggests that recursion alone is not the main driver of RLM's performance; rather, improvements stem from programmatic context interaction. When guided by self-reflection, these programs provide a more reliable way to navigate and reason over long contexts. Overall, SRLM yields more consistent gains across settings, including shorter contexts where RLM can degrade performance, and on semantically demanding tasks requiring deeper contextual understanding.

Additionally, we employ a simple form of self-reflection based on intrinsic uncertainty signals to guide programmatic context interaction. While effective, this design remains limited and leaves room for future work. Future research could explore richer self-reflection mechanisms and tighter integration with decision-making to enable earlier termination and reduce token usage. We hope these findings highlight the importance of context-interaction programming and suggest that leveraging self-reflective signals is a promising direction for improving long-context capabilities in language models.

AI USE STATEMENT

In this work, we used generative AI tools for rewriting sentences and making sure content fits within ICLR 2027 9 page guideline. We have not used generative AI tools for writing python codes within RLM, however we used generative AI tools for generating bash scripts for running agents. We have reviewed all AI-assisted work. Making sure, the content doesn't violate any rules from author guidelines. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Section 2 describes SRLM, including the three uncertainty signals and the joint uncertainty-guided selection rule. Section 3.3 lists the backbone models, the number of candidate trajectories, execution time limits, step and generation-length limits, and API settings used in all experiments. Appendix B describes the benchmarks and evaluation configurations, Appendices C.1 and C.2 give the complete verbalized-confidence elicitation prompt and evaluation judge prompt, and Appendices D.1 and D.2 report detailed per-context-length and ablation results. We have used the rlm-minimal codebase from Zhang et al. (2025a) for running RLM.

REFERENCES

Y Bai, S Tu, J Zhang, H Peng, X Wang, X Lv, S Cao, J Xu, L Hou, Y Dong, et al. Longbenchv2: Towards deeper understanding and reasoning on realistic long-context multitasks. *ArXiv, abs/2412.15204, 2024b. URL https://api. semanticscholar. org/CorpusID*, 274859535, 2024a.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. Longbench: A bilingual, multitask benchmark for long context understanding, 2024b. URL https://arxiv.org/abs/2308.14508.

Marthe Ballon, Andres Algaba, and Vincent Ginis. The relationship between reasoning and performance in large language models–o3 (mini) thinks harder, not longer. *arXiv preprint arXiv:2502.15631*, 2025.

Amanda Bertsch, Adithya Pratapa, Teruko Mitamura, Graham Neubig, and Matthew R Gormley. Oolong: Evaluating long context reasoning and aggregation capabilities. *arXiv preprint arXiv:2511.02817*, 2025.

Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. Longlora: Efficient fine-tuning of long-context large language models. *arXiv preprint arXiv:2309.12307*, 2023.

Zijian Chen, Xueguang Ma, Shengyao Zhuang, Ping Nie, Kai Zou, Andrew Liu, Joshua Green, Kshama Patel, Ruoxi Meng, Mingyi Su, et al. Browsecomp-plus: A more fair and transparent evaluation benchmark of deep-research agent. *arXiv preprint arXiv:2508.06600*, 2025.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready ai agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*, 2025.

Tri Dao and Albert Gu. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality. *arXiv preprint arXiv:2405.21060*, 2024.

Siddartha Devic, Charlotte Peale, Arwen Bradley, Sinead Williamson, Preetum Nakkiran, and Aravind Gollakota. Trace length is a simple uncertainty signal in reasoning models. *arXiv preprint arXiv:2510.10409*, 2025.

Yiran Ding, Li Lyna Zhang, Chengruidong Zhang, Yuanyuan Xu, Ning Shang, Jiahang Xu, Fan Yang, and Mao Yang. Longrope: Extending llm context window beyond 2 million tokens. *arXiv preprint arXiv:2402.13753*, 2024.

Yufeng Du, Minyang Tian, Srikanth Ronanki, Subendhu Rongali, Sravan Bodapati, Aram Galstyan, Azton Wells, Roy Schwartz, Eliu A Huerta, and Hao Peng. Context length alone hurts llm performance despite perfect retrieval, 2025. URL https://arxiv.org/abs/2510.05381.

Jinhao Duan, Hao Cheng, Shiqi Wang, Alex Zavalny, Chenan Wang, Renjing Xu, Bhavya Kailkhura, and Kaidi Xu. Shifting attention to relevance: Towards the predictive uncertainty quantification of free-form large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5050–5063, 2024.

Sabri Eyuboglu, Ryan Ehrlich, Simran Arora, Neel Guha, Dylan Zinsley, Emily Liu, Will Tennien, Atri Rudra, James Zou, Azalia Mirhoseini, et al. Cartridges: Lightweight and general-purpose long context representations via self-study. *arXiv preprint arXiv:2506.06266*, 2025.

Yao Fu, Rameswar Panda, Xinyao Niu, Xiang Yue, Hannaneh Hajishirzi, Yoon Kim, and Hao Peng. Data engineering for scaling language models to 128k context. *arXiv preprint arXiv:2402.10171*, 2024.

Yizhao Gao, Zhichen Zeng, Dayou Du, Shijie Cao, Peiyuan Zhou, Jiaxing Qi, Junjie Lai, Hayden Kwok-Hay So, Ting Cao, Fan Yang, et al. Seerattention: Learning intrinsic sparse attention in your llms. *arXiv preprint arXiv:2410.13276*, 2024.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. arxiv. *arXiv preprint arXiv:2312.00752*, 10, 2023.

Michael Hassid, Gabriel Synnaeve, Yossi Adi, and Roy Schwartz. Don't overthink it. preferring shorter thinking chains for improved llm reasoning. *arXiv preprint arXiv:2505.17813*, 2025.

Kelly Hong, Anton Troynikov, and Jeff Huber. Context rot: How increasing input tokens impacts llm performance. Technical report, Chroma Research, July 2025. URL https://research.trychroma.com/context-rot.

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? *arXiv preprint arXiv:2404.06654*, 2024.

Yuxuan Huang, Yihang Chen, Haozheng Zhang, Kang Li, Huichi Zhou, Meng Fang, Linyi Yang, Xiaoguang Li, Lifeng Shang, Songcen Xu, Jianye Hao, Kun Shao, and Jun Wang. Deep research agents: A systematic examination and roadmap, 2025. URL https://arxiv.org/abs/2506.18096.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. Swe-bench: Can language models resolve real-world github issues? *arXiv preprint arXiv:2310.06770*, 2023.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. Swe-bench: Can language models resolve real-world github issues?, 2024. URL https://arxiv.org/abs/2310.06770.

Bowen Jin, Jinsung Yoon, Jiawei Han, and Sercan O Arik. Long-context llms meet rag: Overcoming challenges for long inputs in rag. *arXiv preprint arXiv:2410.05983*, 2024.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation, 2023. URL https://arxiv.org/abs/2302.09664.

Xunhao Lai, Jianqiao Lu, Yao Luo, Yiyuan Ma, and Xun Zhou. Flexprefill: A context-aware sparse attention mechanism for efficient long-sequence inference. *arXiv preprint arXiv:2502.20766*, 2025.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts, 2023. URL https://arxiv.org/abs/2307.03172.

Potsawee Manakul, Adian Liusie, and Mark J. F. Gales. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models, 2023. URL https://arxiv.org/abs/2303.08896.

Yansheng Mao, Yufei Xu, Jiaqi Li, Fanxu Meng, Haotong Yang, Zilong Zheng, Xiyuan Wang, and Muhan Zhang. Lift: Improving long context understanding of large language models through long input fine-tuning. *arXiv preprint arXiv:2502.14644*, 2025.

Sara Vera Marjanović, Arkil Patel, Vaibhav Adlakha, Milad Aghajohari, Parishad BehnamGhader, Mehar Bhatia, Aditi Khandelwal, Austin Kraft, Benno Krojer, Xing Han Lù, et al. Deepseek-r1 thoughtology: Let's think about llm reasoning. *arXiv preprint arXiv:2504.07128*, 2025.

OpenAI. Deep research. https://openai.com/index/introducing-deepresearch/, 2025. AI-powered research assistant tool.

Yuxiao Qu, Matthew YR Yang, Amrith Setlur, Lewis Tunstall, Edward Emanuel Beeching, Ruslan Salakhutdinov, and Aviral Kumar. Optimizing test-time compute via meta reinforcement fine-tuning. *arXiv preprint arXiv:2503.07572*, 2025.

Stephen Robertson and Hugo Zaragoza. *The probabilistic relevance framework: BM25 and beyond*, volume 4. Now Publishers Inc, 2009.

Parshin Shojaee, Iman Mirzadeh, Keivan Alizadeh, Maxwell Horton, Samy Bengio, and Mehrdad Farajtabar. The illusion of thinking: Understanding the strengths and limitations of reasoning models via the lens of problem complexity. *arXiv preprint arXiv:2506.06941*, 2025.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, Akshay Nathan, Alan Luo, Alec Helyar, Aleksander Madry, Aleksandr Efremov, Aleksandra Spyra, Alex Baker-Whitcomb, Alex Beutel, Alex Karpenko, Alex Makelov, Alex Neitz, Alex Wei, Alexandra Barr, Alexandre Kirchmeyer, Alexey Ivanov, Alexi Christakis, Alistair Gillespie, Allison Tam, Ally Bennett, Alvin Wan, Alyssa Huang, Amy McDonald Sandjideh, Amy Yang, Ananya Kumar, Andre Saraiva, Andrea Vallone, Andrei Gheorghe, Andres Garcia Garcia, Andrew Braunstein, Andrew Liu, Andrew Schmidt, Andrey Mereskin, Andrey Mishchenko, Andy Applebaum, Andy Rogerson, Ann Rajan, Annie Wei, Anoop Kotha, Anubha Srivastava, Anushree Agrawal, Arun Vijayvergiya, Ashley Tyra, Ashvin Nair, Avi Nayak, Ben Eggers, Bessie Ji, Beth Hoover, Bill Chen, Blair Chen, Boaz Barak, Borys Minaiev, Botao Hao, Bowen Baker, Brad Lightcap, Brandon McKinzie, Brandon Wang, Brendan Quinn, Brian Fioca, Brian Hsu, Brian Yang, Brian Yu, Brian Zhang, Brittany Brenner, Callie Riggins Zetino, Cameron Raymond, Camillo Lugaresi, Carolina Paz, Cary Hudson, Cedric Whitney, Chak Li, Charles Chen, Charlotte Cole, Chelsea Voss, Chen Ding, Chen Shen, Chengdu Huang, Chris Colby, Chris Hallacy, Chris Koch, Chris Lu, Christina Kaplan, Christina Kim, CJ Minott-Henriques, Cliff Frey, Cody Yu, Coley Czarnecki, Colin Reid, Colin Wei, Cory Decareaux, Cristina Scheau, Cyril Zhang, Cyrus Forbes, Da Tang, Dakota Goldberg, Dan Roberts, Dana Palmie, Daniel Kappler, Daniel Levine, Daniel Wright, Dave Leo, David Lin, David Robinson, Declan Grabb, Derek Chen, Derek Lim, Derek Salama, Dibya Bhattacharjee, Dimitris Tsipras, Dinghua Li, Dingli Yu, DJ Strouse, Drew Williams, Dylan Hunn, Ed Bayes, Edwin Arbus, Ekin Akyurek, Elaine Ya Le, Elana Widmann, Eli Yani, Elizabeth Proehl, Enis Sert, Enoch Cheung, Eri Schwartz, Eric Han, Eric Jiang, Eric Mitchell, Eric Sigler, Eric Wallace, Erik Ritter, Erin Kavanaugh, Evan Mays, Evgenii Nikishin, Fangyuan Li, Felipe Petroski Such, Filipe de Avila Belbute Peres, Filippo Raso, Florent Bekerman, Foivos Tsimpourlas, Fotis Chantzis, Francis Song, Francis Zhang, Gaby Raila, Garrett McGrath, Gary Briggs, Gary Yang, Giambattista Parascandolo, Gildas Chabot, Grace Kim, Grace Zhao, Gregory Valiant, Guillaume Leclerc, Hadi Salman, Hanson Wang, Hao Sheng, Haoming Jiang, Haoyu Wang, Haozhun Jin, Harshit Sikchi, Heather Schmidt, Henry Aspegren, Honglin Chen, Huida Qiu, Hunter Lightman, Ian Covert, Ian Kivlichan, Ian Silber, Ian Sohl, Ibrahim Hammoud, Ignasi Clavera, Ikai Lan, Ilge Akkaya, Ilya Kostrikov, Irina Kofman, Isak Etinger, Ishaan Singal, Jackie Hehir, Jacob Huh, Jacqueline Pan, Jake Wilczynski, Jakub Pachocki, James Lee, James Quinn, Jamie Kiros, Janvi Kalra, Jasmyn Samaroo, Jason Wang, Jason Wolfe, Jay Chen, Jay Wang, Jean Harb, Jeffrey Han, Jeffrey Wang, Jennifer Zhao, Jeremy Chen, Jerene Yang, Jerry Tworek, Jesse Chand, Jessica Landon, Jessica Liang, Ji Lin, Jiancheng Liu, Jianfeng Wang, Jie Tang, Jihan Yin, Joanne Jang, Joel Morris, Joey Flynn, Johannes Ferstad, Johannes Heidecke, John Fishbein, John Hallman, Jonah Grant, Jonathan Chien, Jonathan

Gordon, Jongsoo Park, Jordan Liss, Jos Kraaijeveld, Joseph Guay, Joseph Mo, Josh Lawson, Josh McGrath, Joshua Vendrow, Joy Jiao, Julian Lee, Julie Steele, Julie Wang, Junhua Mao, Kai Chen, Kai Hayashi, Kai Xiao, Kamyar Salahi, Kan Wu, Karan Sekhri, Karan Sharma, Karan Singhal, Karen Li, Kenny Nguyen, Keren Gu-Lemberg, Kevin King, Kevin Liu, Kevin Stone, Kevin Yu, Kristen Ying, Kristian Georgiev, Kristie Lim, Kushal Tirumala, Kyle Miller, Lama Ahmad, Larry Lv, Laura Clare, Laurance Fauconnet, Lauren Itow, Lauren Yang, Laurentia Romaniuk, Leah Anise, Lee Byron, Leher Pathak, Leon Maksin, Leyan Lo, Leyton Ho, Li Jing, Liang Wu, Liang Xiong, Lien Mamitsuka, Lin Yang, Lindsay McCallum, Lindsey Held, Liz Bourgeois, Logan Engstrom, Lorenz Kuhn, Louis Feuvrier, Lu Zhang, Lucas Switzer, Lukas Kondraciuk, Lukasz Kaiser, Manas Joglekar, Mandeep Singh, Mandip Shah, Manuka Stratta, Marcus Williams, Mark Chen, Mark Sun, Marselus Cayton, Martin Li, Marvin Zhang, Marwan Aljubeh, Matt Nichols, Matthew Haines, Max Schwarzer, Mayank Gupta, Meghan Shah, Melody Huang, Meng Dong, Mengqing Wang, Mia Glaese, Micah Carroll, Michael Lampe, Michael Malek, Michael Sharman, Michael Zhang, Michele Wang, Michelle Pokrass, Mihai Florian, Mikhail Pavlov, Miles Wang, Ming Chen, Mingxuan Wang, Minnia Feng, Mo Bavarian, Molly Lin, Moose Abdool, Mostafa Rohaninejad, Nacho Soto, Natalie Staudacher, Natan LaFontaine, Nathan Marwell, Nelson Liu, Nick Preston, Nick Turley, Nicklas Ansman, Nicole Blades, Nikil Pancha, Nikita Mikhaylin, Niko Felix, Nikunj Handa, Nishant Rai, Nitish Keskar, Noam Brown, Ofir Nachum, Oleg Boiko, Oleg Murk, Olivia Watkins, Oona Gleeson, Pamela Mishkin, Patryk Lesiewicz, Paul Baltescu, Pavel Belov, Peter Zhokhov, Philip Pronin, Phillip Guo, Phoebe Thacker, Qi Liu, Qiming Yuan, Qinghua Liu, Rachel Dias, Rachel Puckett, Rahul Arora, Ravi Teja Mullapudi, Raz Gaon, Reah Miyara, Rennie Song, Rishabh Aggarwal, RJ Marsan, Robel Yemiru, Robert Xiong, Rohan Kshirsagar, Rohan Nuttall, Roman Tsiupa, Ronen Eldan, Rose Wang, Roshan James, Roy Ziv, Rui Shu, Ruslan Nigmatullin, Saachi Jain, Saam Talaie, Sam Altman, Sam Arnesen, Sam Toizer, Sam Toyer, Samuel Miserendino, Sandhini Agarwal, Sarah Yoo, Savannah Heon, Scott Ethersmith, Sean Grove, Sean Taylor, Sebastien Bubeck, Sever Banesiu, Shaokyi Amdo, Shengjia Zhao, Sherwin Wu, Shibani Santurkar, Shiyu Zhao, Shraman Ray Chaudhuri, Shreyas Krishnaswamy, Shuaiqi, Xia, Shuyang Cheng, Shyamal Anadkat, Simón Posada Fishman, Simon Tobin, Siyuan Fu, Somay Jain, Song Mei, Sonya Egoian, Spencer Kim, Spug Golden, SQ Mah, Steph Lin, Stephen Imm, Steve Sharpe, Steve Yadlowsky, Sulman Choudhry, Sungwon Eum, Suvansh Sanjeev, Tabarak Khan, Tal Stramer, Tao Wang, Tao Xin, Tarun Gogineni, Taya Christianson, Ted Sanders, Tejal Patwardhan, Thomas Degry, Thomas Shadwell, Tianfu Fu, Tianshi Gao, Timur Garipov, Tina Sriskandarajah, Toki Sherbakov, Tomer Kaftan, Tomo Hiratsuka, Tongzhou Wang, Tony Song, Tony Zhao, Troy Peterson, Val Kharitonov, Victoria Chernova, Vineet Kosaraju, Vishal Kuo, Vitchyr Pong, Vivek Verma, Vlad Petrov, Wanning Jiang, Weixing Zhang, Wenda Zhou, Wenlei Xie, Wenting Zhan, Wes McCabe, Will DePue, Will Ellsworth, Wulfie Bain, Wyatt Thompson, Xiangning Chen, Xiangyu Qi, Xin Xiang, Xinwei Shi, Yann Dubois, Yaodong Yu, Yara Khakbaz, Yifan Wu, Yilei Qian, Yin Tat Lee, Yinbo Chen, Yizhen Zhang, Yizhong Xiong, Yonglong Tian, Young Cha, Yu Bai, Yu Yang, Yuan Yuan, Yuanzhi Li, Yufeng Zhang, Yuguang Yang, Yujia Jin, Yun Jiang, Yunyun Wang, Yushi Wang, Yutian Liu, Zach Stubenvoll, Zehao Dou, Zheng Wu, and Zhigang Wang. Openai gpt-5 system card, 2025. URL https://arxiv.org/abs/2601.03267.

Weiwei Sun, Miao Lu, Zhan Ling, Kang Liu, Xuesong Yao, Yiming Yang, and Jiecao Chen. Scaling long-horizon llm agent via context-folding, 2025. URL https://arxiv.org/abs/2510.11967.

Jiaming Tang, Yilong Zhao, Kan Zhu, Guangxuan Xiao, Baris Kasikci, and Song Han. Quest: Query-aware sparsity for efficient long-context llm inference. *arXiv preprint arXiv:2406.10774*, 2024.

Linwei Tao, Yi-Fan Yeh, Minjing Dong, Tao Huang, Philip Torr, and Chang Xu. Revisiting uncertainty estimation and calibration of large language models, 2025. URL https://arxiv.org/abs/2505.23854.

Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Arne Vanhoyweghen, Brecht Verbeken, Andres Algaba, and Vincent Ginis. Lexical hints of accuracy in llm reasoning chains. *arXiv preprint arXiv:2508.15842*, 2025.

Roger Waleffe, Wonmin Byeon, Duncan Riach, Brandon Norick, Vijay Korthikanti, Tri Dao, Albert Gu, Ali Hatamizadeh, Sudhakar Singh, Deepak Narayanan, et al. An empirical study of mamba-based language models. *arXiv preprint arXiv:2406.07887*, 2024.

Weizhi Wang, Li Dong, Hao Cheng, Xiaodong Liu, Xifeng Yan, Jianfeng Gao, and Furu Wei. Augmenting language models with long-term memory. *Advances in Neural Information Processing Systems*, 36:74530–74543, 2023.

Xingyao Wang, Yangyi Chen, Lifan Yuan, Yizhe Zhang, Yunzhu Li, Hao Peng, and Heng Ji. Executable code actions elicit better llm agents. In *Forty-first International Conference on Machine Learning*, 2024.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia Glaese. Browsecomp: A simple yet challenging benchmark for browsing agents. *arXiv preprint arXiv:2504.12516*, 2025.

Xixi Wu, Kuan Li, Yida Zhao, Liwen Zhang, Litu Ou, Huifeng Yin, Zhongwang Zhang, Xinmiao Yu, Dingchu Zhang, Yong Jiang, et al. Resum: Unlocking long-horizon search intelligence via context summarization. *arXiv preprint arXiv:2509.13313*, 2025.

Miao Xiong, Zhiyuan Hu, Xinyang Lu, Yifei Li, Jie Fu, Junxian He, and Bryan Hooi. Can llms express their uncertainty? an empirical evaluation of confidence elicitation in llms. *arXiv preprint arXiv:2306.13063*, 2023.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*, 2022.

Dongkeun Yoon, Seungone Kim, Sohee Yang, Sunkyoung Kim, Soyeon Kim, Yongil Kim, Eunbi Choi, Yireun Kim, and Minjoon Seo. Reasoning models better express their confidence. *arXiv preprint arXiv:2505.14489*, 2025.

Hongli Yu, Tinghong Chen, Jiangtao Feng, Jiangjie Chen, Weinan Dai, Qiying Yu, Ya-Qin Zhang, Wei-Ying Ma, Jingjing Liu, Mingxuan Wang, et al. Memagent: Reshaping long-context llm with multi-conv rl-based memory agent. *arXiv preprint arXiv:2507.02259*, 2025.

Alex L Zhang, Tim Kraska, and Omar Khattab. Recursive language models. *arXiv preprint arXiv:2512.24601*, 2025a.

Anqi Zhang, Yulin Chen, Jane Pan, Chen Zhao, Aurojit Panda, Jinyang Li, and He He. Reasoning models know when they're right: Probing hidden states for self-verification. *arXiv preprint arXiv:2504.05419*, 2025b.

Guibin Zhang, Muxin Fu, Guancheng Wan, Miao Yu, Kun Wang, and Shuicheng Yan. G-memory: Tracing hierarchical memory for multi-agent systems. *arXiv preprint arXiv:2506.07398*, 2025c.

Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu, Mengmeng Ji, Hanchen Li, Urmish Thakker, James Zou, and Kunle Olukotun. Agentic context engineering: Evolving contexts for self-improving language models, 2025d. URL https://arxiv.org/abs/2510.04618.

Liang Zhao, Tianwen Wei, Liang Zeng, Cheng Cheng, Liu Yang, Peng Cheng, Lijie Wang, Chenxia Li, Xuejie Wu, Bo Zhu, et al. Longskywork: A training recipe for efficiently extending context length in large language models. *arXiv preprint arXiv:2406.00605*, 2024.

A Extended Related Works

Context Window Expansion A widely adopted approach for handling long contexts in language models has been to increase the maximum context window through architectural modifications or training strategies. For example, advancements in positional encoding and attention scaling have enabled models to process substantially longer inputs (Chen et al., 2023; Ding et al., 2024; Mao et al., 2025). Beyond these techniques, several architectural directions have been explored to further improve long-context scalability, including sparsity-based mechanisms (Tang et al., 2024; Gao et al., 2024; Lai et al., 2025), state-space models (Gu & Dao, 2023; Dao & Gu, 2024; Waleffe et al., 2024), retrieval-augmented fine-tuning (Jin et al., 2024; Wang et al., 2023), and key–value (KV) cache compression techniques (Eyuboglu et al., 2025). Collectively, these methods aim to improve the scalability of language models with respect to input length and enable effective yet efficient inference over long contexts. Despite these advances, extending the context window alone does not fully address the challenges of long-context reasoning. Recent empirical studies show that model performance remains constrained by the effective context length. For example, LongBench-v2 (Bai et al., 2024a) finds that frontier models with extended context windows still struggle on realistic long-context multitask benchmarks, achieving only modest improvements once input lengths exceed certain context thresholds. These results suggest that scaling the context window, while helpful, is insufficient to guarantee robust reasoning over very large contexts in real-world settings.

Agentic Long-Context Approaches. There has been an alternative line of research to complement earlier long-context handling approaches with inference-time strategies. Recent works treat long-context as a procedural and agentic problem, leveraging LLMs as tools invoked through programmatic strategies that iteratively interact with context. For example, Recursive Language Model (RLM) (Zhang et al., 2025a) utilize this idea by treating the context and input prompt as a variable in an external programming environment that can be decomposed, queried, and recursively processed through program execution. RLMs show significant benefits over monolithic prompting and context summarization methods on several long-context benchmarks, demonstrating the effectiveness of programmatic context interaction. Related approaches include code-execution agents, such as CodeAct (Wang et al., 2024) which enable iterative code generation and execution for flexible search over context, as well as summarization-based agents such as ReSum (Wu et al., 2025) that periodically compress interaction histories in agentic web search. There are also existing works on memory-augmented agents that introduce explicit long-term storage to better extract and retrieve salient information across long interactions, e.g., Mem0 (Chhikara et al., 2025) and G-Memory (Zhang et al., 2025c). While these agentic and programmatic approaches to long-context tasks show promise, their decisions about what to read, summarize, or revisit typically rely on surface-level heuristics rather than semantic understanding. Consequently, they perform better at structurally localized tasks but struggle with semantically dense ones requiring deep comprehension more than search.

Long-Context Evaluation. A growing body of work in literature has focused on evaluating LLMs under long-context settings. Benchmarks such as LongBench (Bai et al., 2024b), and its successor LongBench v2 (Bai et al., 2024a) evaluate realistic tasks involving long documents, dialogues, and codebases, revealing substantial performance degradation as context length increases. Complementary benchmarks emphasize different stressors. For example, Single Needle in a Haystack (S-NIAH) (Hsieh et al., 2024) tasks test retrieval robustness at extreme lengths, where most of the context is irrelevant. In contrast, benchmarks like OOLONG (Bertsch et al., 2025) explicitly target long-context reasoning and aggregation, requiring models to process and combine information across all parts of the long iputs. BrowseComp (Wei et al., 2025) and BrowseComp-Plus (Chen et al., 2025) also further extend the long-context evaluation to agentic and deep-research settings, where models must persistently navigate, remember, and integrate information across many documents or browsing steps. Collectively, these benchmarks show that long-context reasoning remains a ubiquitous challenge for frontier models across a wide range of diverse and practically important tasks.

Confidence Estimation in Large Language Models. Estimating model confidence is an important component for enabling reliable reasoning and self-correction in language models. A growing body of work studies how uncertainty signals can be extracted from LLMs without requiring additional fine-tuning. One line of work leverages multiple samples from the model's predictive distribution

to estimate uncertainty via agreement across generations, commonly referred to as sampling-based confidence or self-consistency (Wang et al., 2022; Kuhn et al., 2023; Manakul et al., 2023). In this setting, the empirical frequency of an answer across sampled outputs provides a natural estimate of the model's confidence in that prediction. Another direction investigates the model's ability to explicitly report its own uncertainty. Several recent studies evaluate this called as verbalized confidence, where the model is prompted to provide calibrated confidence scores alongside its predictions (Xiong et al., 2023; Yoon et al., 2025). Empirical comparisons suggest that self-verbalized confidence can serve as a strong and more semantic zero-shot uncertainty estimator and often outperforms approaches based solely on token probabilities or sampling statistics (Tao et al., 2025). Other approaches estimate uncertainty using internal model signals such as token likelihoods (Duan et al., 2024) or learned probes over hidden representations (Zhang et al., 2025b). More recently, work in the reasoning literature has identified behavioral signals in the generation process of recent frontier models that correlate with model uncertainty. In particular, several studies observe that incorrect reasoning trajectories tend to be longer and more deliberative than correct ones (Marjanović et al., 2025; Ballon et al., 2025). This phenomenon suggests that reasoning trace length can serve as an implicit proxy for epistemic uncertainty (Devic et al., 2025; Vanhoyweghen et al., 2025). While prior work has primarily exploited this observation to improve reasoning efficiency or reduce unnecessary deliberation (Qu et al., 2025; Hassid et al., 2025), it also provides a useful signal for identifying reliable reasoning trajectories. Our work builds on these insights and leverages multiple complementary uncertainty signals, including sampling-based consistency, verbalized confidence, and behavioral indicators from reasoning lengths, to guide uncertainty-aware self-reflection during program search for long context interaction.

B Details on Datasets

BrowseComp-Plus (Chen et al., 2025) is a controlled evaluation benchmark for "deep research" agents: systems that combine LLM reasoning with search/retrieval tools to answer complex, reasoning-intensive, fact-seeking questions that require combining evidence across multiple sources. Unlike evaluations that rely on live, opaque web search APIs, BrowseComp-Plus fixes the document corpus, enabling reproducible experiments and clearer separation between retrieval failures and reasoning failures. BrowseComp-Plus is derived from BrowseComp (Wei et al., 2025), a benchmark of challenging short-answer browsing problems originally released by OpenAI. BrowseComp was intentionally designed around short, verifiable answers to keep grading straightforward, even though the search process may be difficult. The BrowseComp-Plus dataset is publicly available[1] with queries, and the document relevance judgments, including query text and per-query lists of evidence, gold, and negative documents. To reduce benchmark leakage into training corpora of frontier language models and discourage copying benchmark content, the dataset release is obfuscated (with query_id as the only non-obfuscated field), and includes explicit anti-leakage canary mechanisms. To use the dataset, we need to first decrypt the dataset as suggested here.

In our experiments, we use the "1K documents" evaluation setup, following (Zhang et al., 2025a): for each question, instead of performing live retrieval, we provide 1,000 randomly selected documents as the long context, with the guarantee that the question's gold and evidence documents exist within that subset; we then evaluate on 150 randomly sampled questions and report accuracy (percentage of correct final answers). The distribution of context length in terms of tokens across these 150 questions are provided in Figure 6. In our setup, we compute robust accuracy using an LLM-as-Judge framework. Instead of relying on exact string matching, the judge compares the model's generated answer with the provided ground truth using an evaluation prompt (see Section C.2), allowing for semantically equivalent or string variations of an answer.

OOLONG (Bertsch et al., 2025) is a long-context benchmark designed to evaluate large-scale information aggregation. It is motivated by the observation that many long-context benchmarks reduce to sparse retrieval tasks (i.e., needle-in-a-haystack settings where most of the input is irrelevant noise). In contrast, many real-world long-context tasks require processing a substantial portion of the context and aggregating numerous small decisions to produce a final answer. OOLONG benchmark is divided into two categories: OOLONG-synth, which consists of "naturalistic synthetic" tasks derived from existing in-context learning classification datasets and enables controlled analysis of

[1] https://huggingface.co/datasets/Tevatron/browsecomp-plus

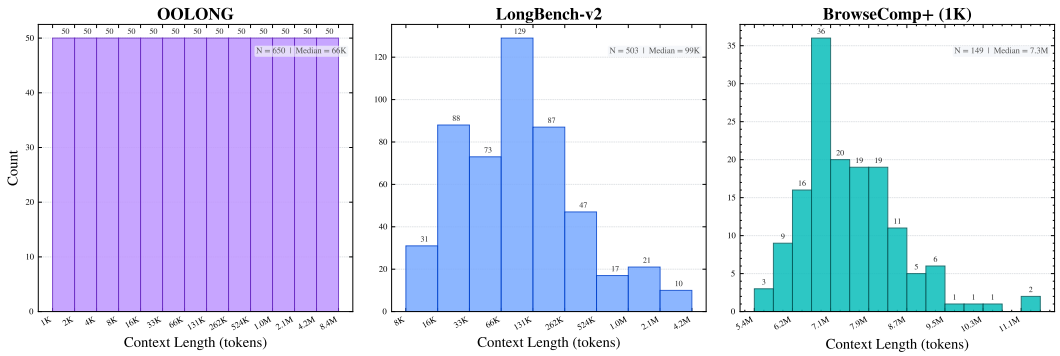


Figure 6: Distribution of input context lengths across different benchmarks: OOLONG, LongBench-v2, and BrowseComp-Plus.

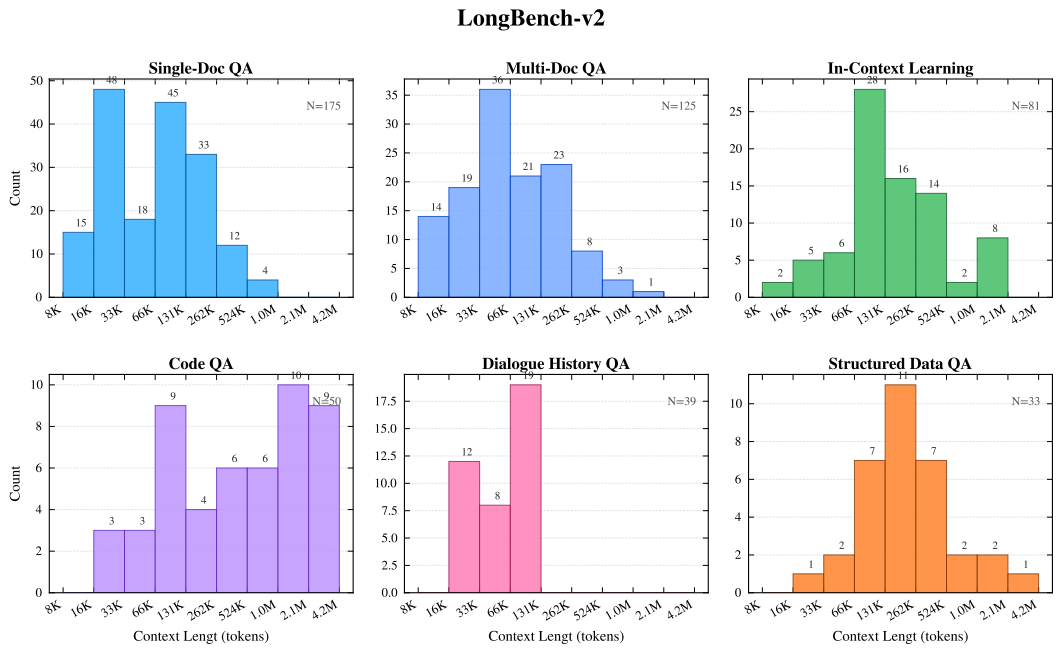


Figure 7: Context length distributions for different task categories in LongBench-v2 benchmark dataset.

difficulty factors; and OOLONG-real, which contains downstream aggregation questions over long-form conversational transcripts, designed to be less easily decomposed into independent in-context learning examples.

In our experiments, we focus on the OOLONG-synth `trec_coarse` configuration, following (Zhang et al., 2025a). We evaluate 650 tasks across context lengths ranging from 1K to 8M tokens (50 tasks per context length as shown in Figure 6) and report accuracy following the benchmark's original scoring protocol, including exponential partial credit for numeric outputs. For categorical outputs (e.g., labels, dates, user IDs, or comparisons), correctness is determined by exact match with the ground truth or by judge-based equivalence assessment (with same prompt from Section C.2). For numeric outputs, the score is computed as $0.75^{|y-\hat{y}|}$ which assigns partial credit as predictions approach the true value. We use the publicly available dataset [2] together with the official open-source code[3] for experiments.

[2] https://huggingface.co/datasets/oolongbench/oolong-synth

[3] https://github.com/abertsch72/oolong

LongBench-v2 (Bai et al., 2024a) is an extended version of the LongBench benchmark (Bai et al., 2024b), designed to evaluate the ability of large language models to perform reasoning and comprehension over diverse realistic long contexts. To make evaluation more robust and less sensitive to generation formatting, the benchmark standardizes tasks into multiple-choice questions (A/B/C/D), allowing consistent comparison across models with different behaviors. LongBench-v2 contains 503 evaluation instances with context lengths ranging from 8K to 4M tokens. The overall context length distribution of this dataset is shown in Figure 6. The benchmark spans a wide spectrum of realistic long-context scenarios and is organized into six task categories: single-document question answering, multi-document question answering, long in-context learning, dialogue history understanding, code repository understanding, and long structured data understanding. These categories collectively capture a diverse set of long-context reasoning patterns, including extracting from and comprehending lengthy documents, synthesizing and finding evidence across multiple sources, reasoning over extended conversational histories, understanding large software repositories, and querying or aggregating information from large structured tables. The detailed context length distributions for each of these domain categories are provided in Figure 7.

To assess how different long-context methods generalize across diverse domains, we evaluate on all task categories of the LongBench-v2 dataset in our experiments. This differs from prior work (Zhang et al., 2025a), which focuses only on the CodeQA subset of the benchmark. Using the full benchmark enables a more comprehensive evaluation of long-context capability across heterogeneous domains and task semantics. In our experiments, we use the official LongBench-v2 dataset which is publicly available[4] together with the benchmark's open-source code for evaluation[5]. Simialr to other datasets, to evaluate correctness of answers in this benchmark, we also use the judge with prompt as in Section C.2.

C Prompt Details

C.1 Verbalized Confidence Elicitation

At each intermediate generation step t of every candidate program $p^{(k)}$, we elicit the model's self-assessed confidence by appending a fixed structured instruction to the generation prompt. The instruction is designed to (i) require the model to produce a parseable confidence score, (ii) enforce a consistent format across all steps and all programs, and (iii) encourage nuanced and calibrated self-assessment rather than coarse or overconfident reporting. The following instruction is appended verbatim to the model's prompt for every generation. For the rare cases that the verbalized confidence is not provided by model, the average of verbalized confidence of other steps in the same trajectory is used to fill the gap.

Confidence Prompt

```
IMPORTANT: At the end of your response, you MUST include
your confidence in your answer using this exact JSON format
on a new line:
```json
{"confidence": <number between 0.000 and 100.000
                with up to 3 decimal points>}
```
Be precise and nuanced in your confidence assessment.
```

C.2 Evaluation Judge

To compute final task accuracy across all datasets used in our experiments, we employ an LLM-as-Judge evaluation procedure. Many long-context benchmarks contain answers that may appear in slightly different textual forms while still being semantically correct (e.g., variations in phrasing, formatting differences, or additional explanatory details). Relying solely on exact string matching can therefore incorrectly penalize some of the valid and semantically equivalent answers. To address this issue, we use an automated judge model with the prompt below that compares the model-generated

[4] https://huggingface.co/datasets/zai-org/LongBench-v2
[5] https://github.com/THUDM/LongBench

responses with the ground-truth answer and determines correctness based on semantic equivalence rather than strict lexical matching.

Grading Prompt

```
Judge whether the following [response] to [question] is correct or not
based on the precise and unambiguous [correct_answer] below.

[question]: {QUESTION}

[response]: {RESPONSE}

[correct_answer]: {CORRECT_ANSWER}

Your judgement must be in the format and criteria specified below:

extracted_final_answer: The final exact answer extracted from the [response].

correct_answer: Repeat the [correct_answer] given above.

reasoning: Explain why the extracted_final_answer is correct or incorrect based on
[correct_answer], in the context of this [question]. You should judge
whether the extracted_final_answer is semantically equivalent to
[correct_answer], allowing the extracted_final_answer to be string
variations of [correct_answer]. You should also allow the
extracted_final_answer to be more precise or verbose than
[correct_answer], as long as its additional details are correct.
Do not comment on any background to the problem, do not attempt to
solve the problem, do not argue for any answer different than
[correct_answer], focus only on whether the answers are
semantically equivalent.

correct: Answer 'yes' if extracted_final_answer matches the [correct_answer]
given above, or is within a small margin of error for numerical
problems. Answer 'no' otherwise, i.e. if there is any inconsistency,
ambiguity, non-equivalency, or if the extracted answer is incorrect.

confidence: The extracted confidence score between 0|%| and 100|%| from [response].
```

D Additional Results

D.1 Detailed Results Across Context Lengths

Improvement over the base model. In this section, we provide a more detailed view of the context-length experiments summarized in Section 3.5. While the main paper presents aggregated results for comparison to base across context splits and task domains in LongBench-v2 dataset, Figure 8-11 show performance trends at a finer granularity across context-length bins and different task categories. These additional results help illustrate how different approaches behave as the input context grows from moderately long inputs to ver long multi-million token sequences. Figures 8 and 9 present the detailed improvement relative to the base model (Δ vs. base) for each context-length bin on the OOLONG and LongBench-v2 benchmarks, respectively. Each bin contains approximately the same number of evaluation instances to ensure fair comparison across context scales. The plots report the performance difference between each method and the base model, allowing us to directly observe whether a method improves or degrades performance at different context lengths. As it can be shown, the context length is separated into contexts within (<131K) and near/beyond ($\geq$ 131K) the model's context window with the highlighted yellow background.

Consistent with the trends observed in Figure 2 in the main paper, the advantage of SRLM becomes more pronounced at longer contexts. In both benchmarks and across both backbone models, the improvement over the base model increases as context length grows. In contrast, RLM mostly underperforms the base model at shorter context lengths, indicating that recursive decomposition may introduce unnecessary overhead when the context already fits within the model's window.

Task-domain analysis. To further understand how long-context methods behave across different types of reasoning tasks, Figures 10 and 11 provide detailed breakdowns by task category for the context analysis performance on LongBench-v2 benchmark. These plots report both the absolute accuracy and the improvement relative to the base model across context lengths detailed for each

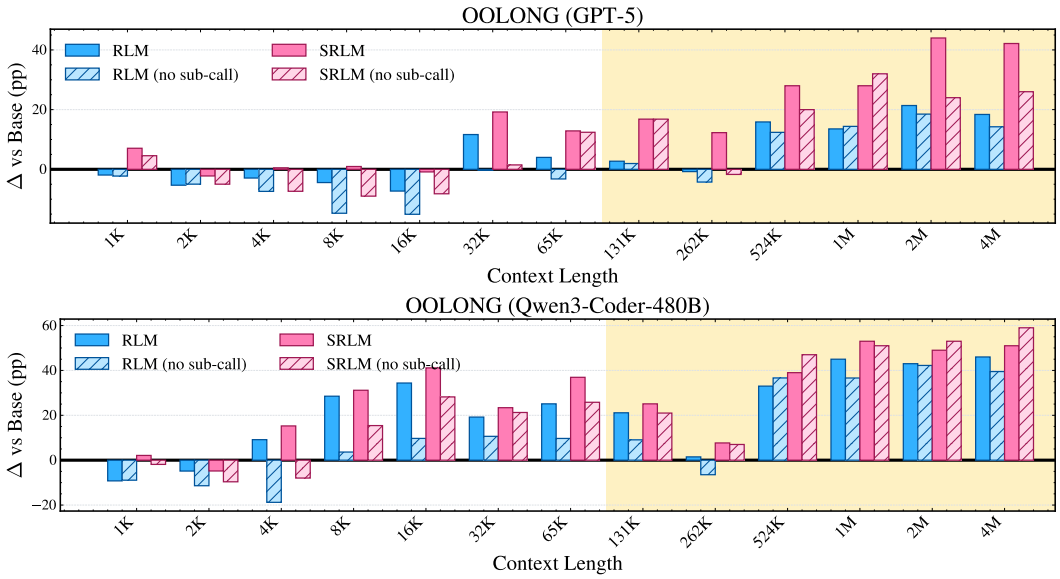


Figure 8: Performance gains over the base model across context lengths on the OOLONG benchmark. Bars show accuracy change vs. base for each context-length bin. Results are shown for GPT-5 (**top**) and Qwen3-Coder-480B (**bottom**). The shaded yellow region marks contexts near or beyond the context window limits ($\geq$ 131K tokens).

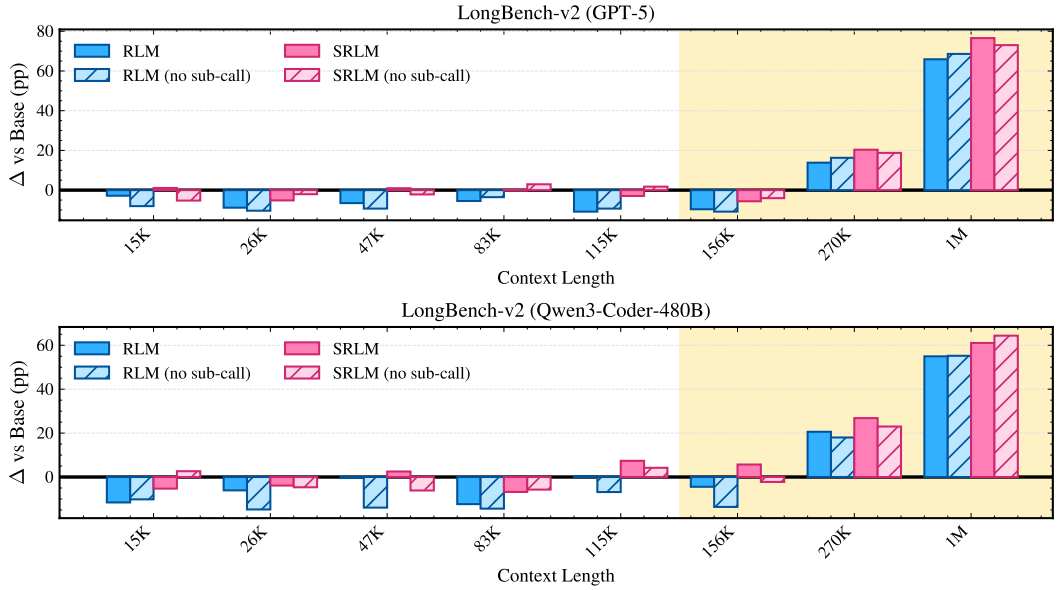


Figure 9: Performance gains over the base model across context lengths on the LongBench-v2 benchmark. Bars show accuracy change vs. base for each context-length bin with balanced samples. Results are shown for GPT-5 (**top**) and Qwen3-Coder-480B (**bottom**) backbones. The shaded yellow region marks contexts near or beyond the context window limits ($\geq$ 131K tokens).

task domain. The results reveal that the benefits self-reflection in SRLM are consistent across diverse tasks, including code repository understanding, single-document question answering, multi-document question answering, and long in-context learning. In particular, improvements become more pronounces once the context length approaches or exceeds the context window limits of the underlying model (shaded yellow region).

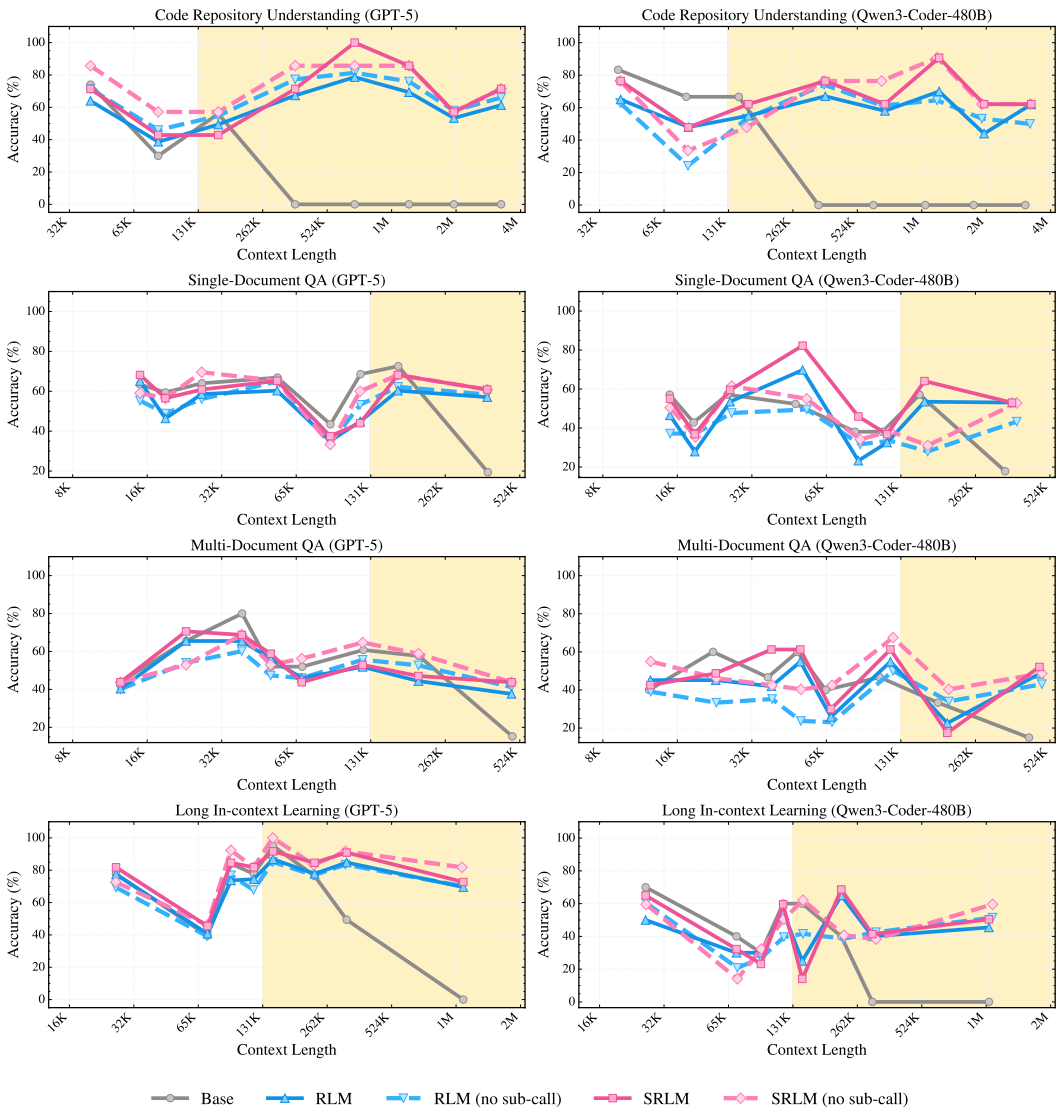


Figure 10: Accuracy versus context length across LongBench-v2 task categories. Results for GPT-5 (**left**) and Qwen3-Coder-480B (**right**) backbones, with each subplot showing performance across context scales for a specific task domain.

D.2 Detailed Ablation Results

This section provides a detailed breakdown of the ablation experiments summarized in Section 3.8. While the main paper reports aggregated results across recursive/nonrecursive runs and backbones, the figures in this appendix present the results separately for each backbone model, dataset, and execution setting. These analyses allow us to better understand how individual uncertainty signals contribute to the performance of SRLM under different conditions.

Figure 12 shows the detailed analysis for contribution of each uncertainty signal used in the self-reflection mechanism of SRLM. The results are reported separately for GPT-5 and Qwen3-Coder-480B backbones, as well as for runs with recursive sub-calls and runs without sub-calls. Across all datasets and model backbones, the full SRLM configuration consistently achieves the best performance. Variants that rely on individual uncertainty signals alone show smaller improvements compared to the combination in SRLM for all the queries with context length $\geq$ 131K. This observation confirms that the three uncertainty signals capture complementary aspects of model uncertainty

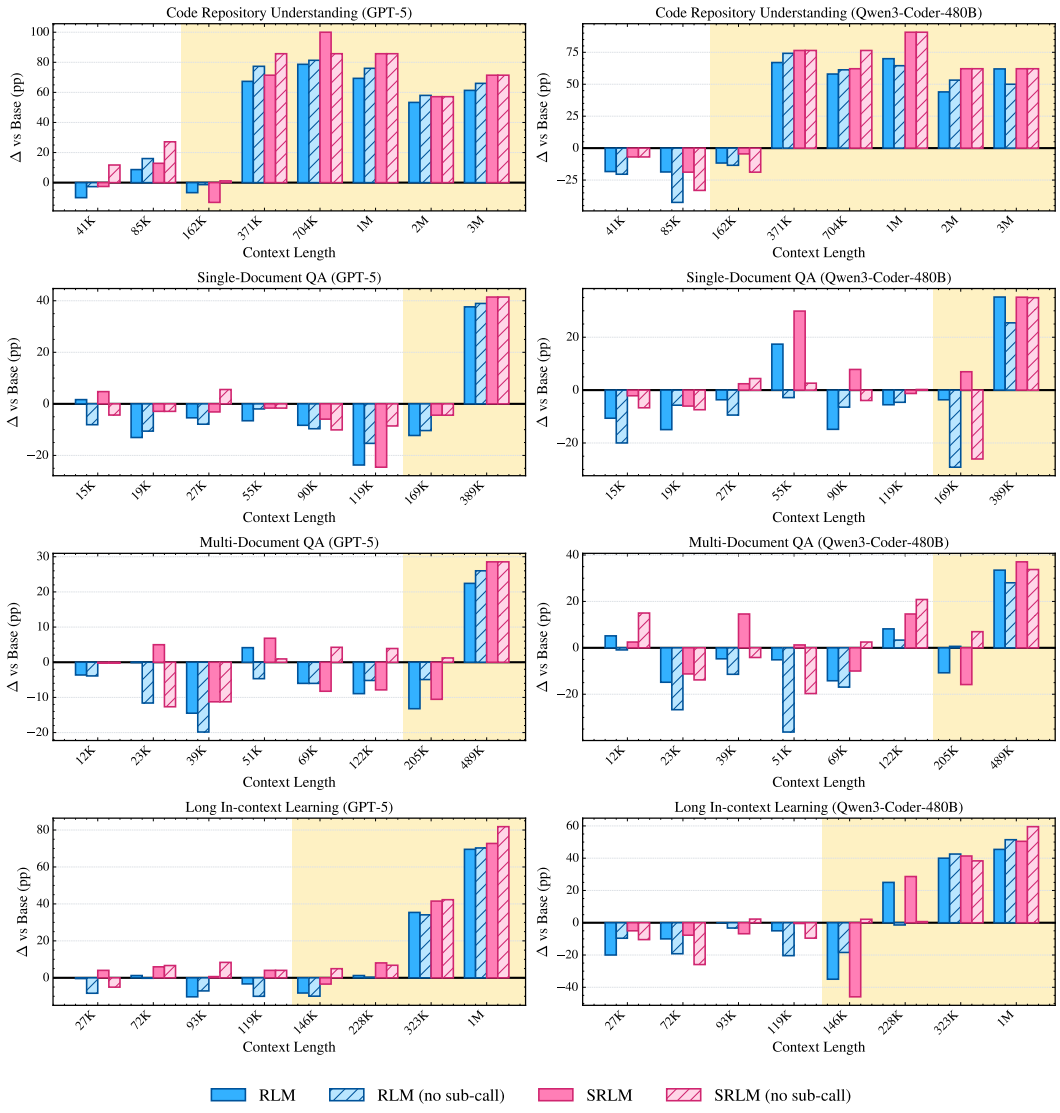


Figure 11: Improvement over the base model across context lengths for LongBench-v2 task domains. Results for GPT-5 (**left**) and Qwen3-Coder-480B (**right**), with each subplot showing performance change across context scales for a specific task domain.

during reasoning and this is true across all backbones and runs. The detailed plots also reveal that the relative contribution of each signal can vary depending on the backbone model and dataset. For example, reasoning trace length tends to provide stronger signals in some datasets, whereas verbalized confidence tends to be more informative on others when the model's internal calibration is reliable. However, across all settings, combining the signals through the self-reflection mechanism consistently provides the best performance.

Figure 13 further analyzes the interaction between fine-grained signals of verbalized confidence and reasoning trace length. For this analysis, predictions are grouped into bins with approximately equal numbers of samples based on their confidence scores and reasoning trace lengths. Each heatmap cell reports the empirical accuracy obtained for a given combination of these two signals. The heatmaps reveal several interesting patterns. For example, the relation between verbalized confidence and accuracy is not strictly monotonic, i.e., predictions with high confidence do not always correspond to correct answers. Similarly, shorter or longer reasoning traces alone do not reliably indicate correctness. Instead, the highest accuracy regions often appear when both signals align and their

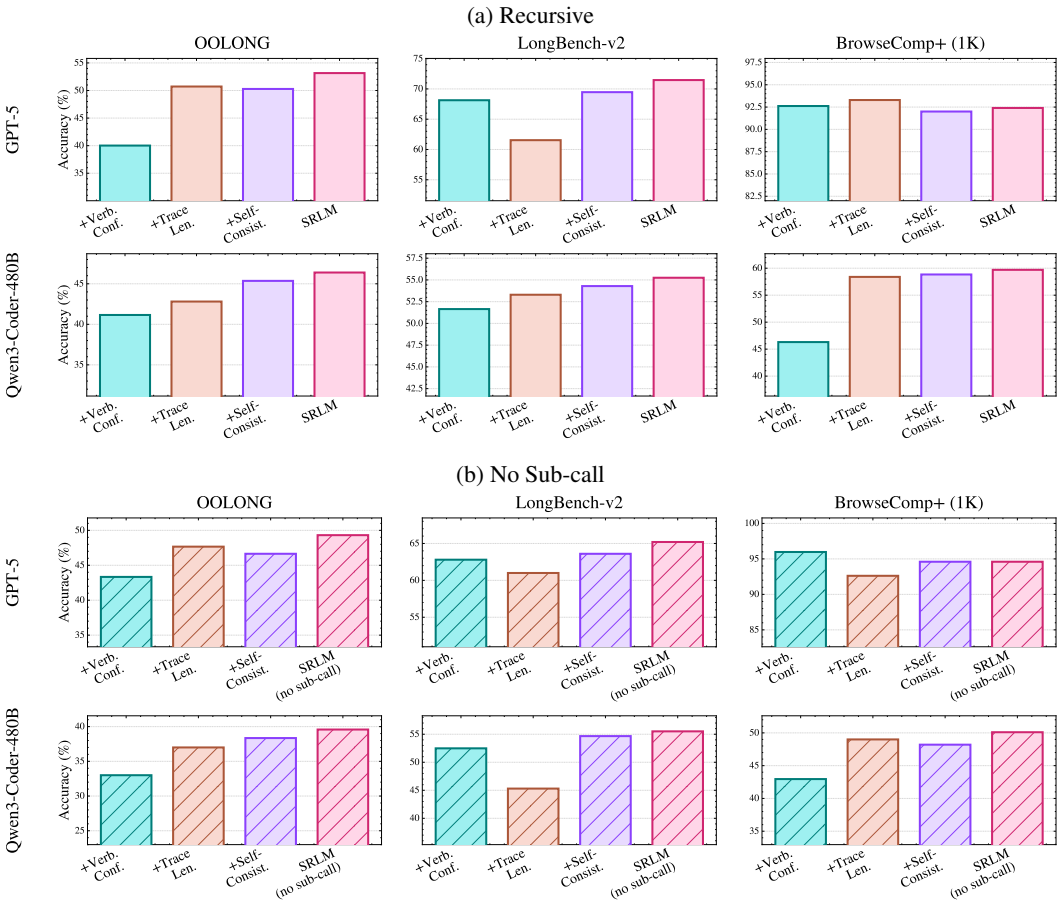


Figure 12: Ablation of uncertainty signals in SRLM 's self-reflection mechanism. Results for GPT-5 and Qwen3-Coder-480B backbones on OOLONG, LongBench-v2, and BrowseComp+ (1K) benchmarks with context $\geq$ 131K. **(a) With** recursive sub-calls; **(b) Without** recursive sub-calls.

joint combination is considered. By jointly considering semantic uncertainty (verbalized confidence) and behavioral uncertainty (reasoning trace characteristics), SRLM is able to provide more reliable self-reflection for context-interaction programs in the scope of long-context.

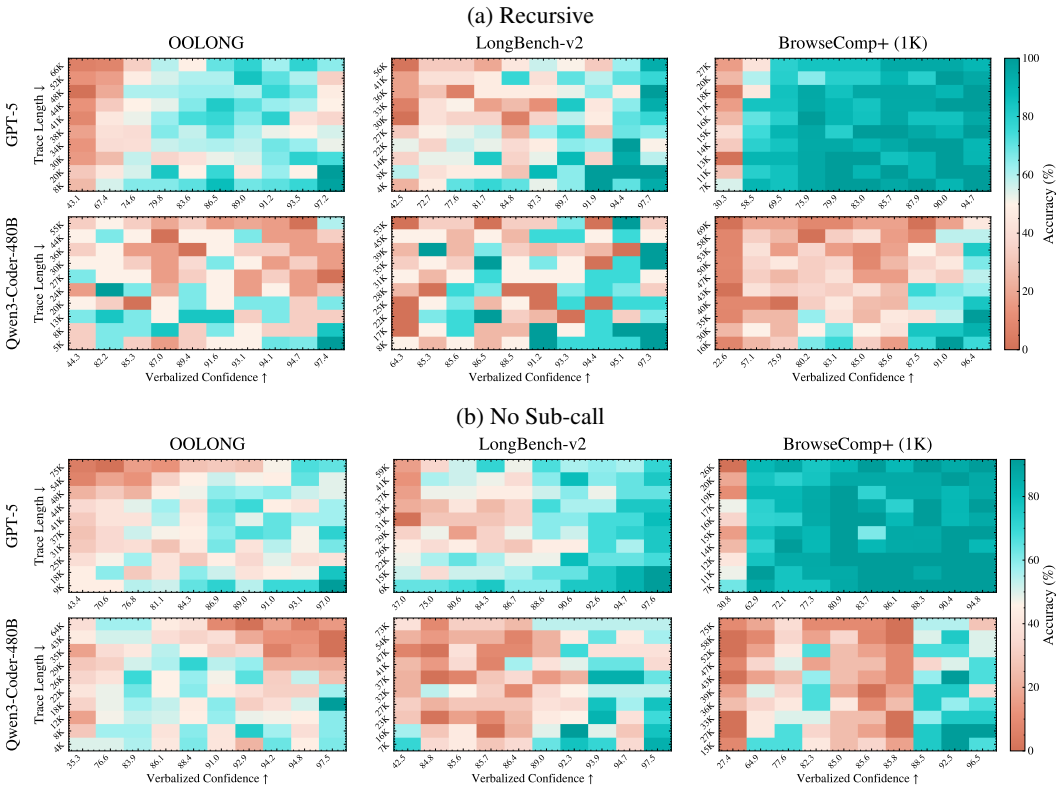


Figure 13: Analysis of the relationship between verbalized confidence, reasoning length, and accuracy. Heatmaps show accuracy across equal-sample bins of confidence (x-axis) and reasoning length (y-axis). Results are reported for GPT-5 and Qwen3-Coder-480B backbones on the OOLONG, LongBench-v2, and BrowseComp+ (1K) benchmarks with context $\geq$ 131K. **(a) With** recursive sub-calls; **(b) Without** recursive sub-calls.