SciTrek: Evaluating and Improving Long-Context Numerical Reasoning over Scientific Articles

Miao Li, Alexander Gurung, Irina Saparina, Mirella Lapata
School of Informatics
University of Edinburgh
miao.li@ed.ac.uk, alex.gurung@ed.ac.uk, i.saparina@sms.ed.ac.uk, mlap@inf.ed.ac.uk

arXiv:2509.21028v6 [cs.AI] 26 Aug 2026

Abstract

We introduce SciTrek, a synthetic question-answering dataset for assessing and improving long-context numerical reasoning in large language models (LLMs). Existing long-context datasets with inputs beyond 64K tokens either target simple information retrieval or, when they do involve reasoning, rely on artificial contexts, while numerical reasoning remains largely overlooked in both cases. SciTrek addresses these limitations with questions that require numerical operations (e.g., counting, sorting, aggregation, and comparison) over collections of full-text scientific articles. Questions are generated automatically by formulating them as SQL queries over a database of article metadata (titles, authors, and references), and ground-truth answers are obtained by executing these queries. The underlying SQL provides a transparent, verifiable specification of the reasoning each question requires, enabling fine-grained error analysis. Furthermore, the proposed fully automated pipeline scales to arbitrary context lengths and dataset sizes with minimal human supervision, mitigating data contamination and supplying abundant data for post-training. Extensive experiments show that frontier open-weight and proprietary LLMs struggle even with ostensibly simple questions: the best-performing model achieves only 46.5% exact match at 128K tokens, and performance degrades steadily as contexts grow. Fine-grained analysis reveals systematic weaknesses on citation-related questions and on compound logical conditions, particularly those involving negation. Finally, post-training open-weight models on SciTrek improves their numerical reasoning in ways that generalise to out-of-domain long-context tasks.[1]

1 Introduction

Large language models (LLMs) now routinely support context windows of 128K tokens, with some exceeding 1M tokens (Gemini Team, 2025; OpenAI, 2025; Yang et al., 2025).[2] However, models that accommodate such inputs do not necessarily reason over them reliably, struggling, for instance, to integrate information scattered across multiple documents (Bai et al., 2025; Wang et al., 2024c). Diagnosing and addressing these failures requires data: progress in long-context modelling has consistently depended on large, high-quality datasets for both evaluation and training (Liu et al., 2025). While a growing number of such datasets have been developed (Kuratov et al., 2024; Maekawa et al., 2025; Bai et al., 2025; Yen et al., 2025), existing efforts suffer from three key limitations.

First, although the inputs are sufficiently long, most existing datasets emphasize simple information retrieval rather than multi-step reasoning. Needle-In-A-Haystack tasks (Kamradt, 2023; Hsieh et al., 2024), for instance, have been largely saturated by state-of-the-art long-context language models (Gemini Team, 2025; OpenAI, 2025; Yang et al., 2025). Second, datasets that target reasoning face a tradeoff between naturalness and scalability. Some rely on artificial contexts created by injecting noise into short passages (Kuratov et al., 2024) (e.g., HotpotQA; Yang et al. 2018) or by synthesizing inputs with LLMs (Maekawa et al., 2025; Wang et al., 2024b), which limits ecological validity. Other datasets are built on natural long contexts but depend

[1] Our dataset and code are available at https://github.com/oaimli/SciTrek.
[2] In this paper, K denotes 1,024 tokens and M denotes 1,024K.

Question	SQL Query	Answer
What is the highest number of authors that any single article has?	`SELECT MAX(author_count) FROM articles`	10
What is the word count of the titles of articles, sorted by the number of authors in ascending order?	`SELECT title_word_count FROM articles ORDER BY author_count ASC`	9, 17, 5, 6, 9, 12
How many references do articles with exactly two authors have?	`SELECT reference_count FROM articles WHERE author_count = 2`	16
What is the total number of words in the titles of all articles that have exactly 60 references?	`SELECT SUM(title_word_count) FROM articles WHERE reference_count = 60`	13
What are the names of the authors who are either the first or second author of an article, listed in descending order of their position?	`SELECT author_name FROM article_author WHERE author_position < 2 ORDER BY author_position DESC`	N. Shazeer, A. Vaswani
How many articles are cited by other articles but do not cite any other articles?	`SELECT COUNT(*) FROM articles WHERE article_id NOT IN (SELECT article_id_citing FROM citing_cited) AND article_id IN (SELECT article_id_cited FROM citing_cited)`	2

Table 1: The SciTrek dataset: example questions with corresponding SQL queries and answers.

on substantial human annotation, which makes scaling to rapidly expanding context windows difficult and increases the risk of data contamination and obsolescence (Asai et al., 2024; An et al., 2024; Wang et al., 2024c; Bai et al., 2025). Third, numerical reasoning — counting, sorting, aggregation, and comparison — remains largely unexplored in long-context datasets, despite being fundamental to real-world tasks (Wolfson et al., 2020; 2025). Moreover, because existing datasets rarely make explicit which reasoning skills each question targets, it is difficult to attribute model failures to specific capabilities.

In this paper, we introduce SciTrek, a question-answering dataset over scientific articles that addresses all three limitations. Scientific article collections provide a particularly suitable testbed for long-context numerical reasoning: they contain rich relational structure through citations, authorship, and cross-references, offer naturally occurring inputs, and scale seamlessly to long contexts as their size grows. We generate questions requiring numerical operations, including counting, sorting, aggregation, and comparison, over core elements of scientific articles: titles, authors, and references. By relying on domain-general, readily identifiable elements rather than scientific content, the resulting tasks minimize dependence on domain-specific understanding while directly targeting long-context numerical reasoning itself. Because these operations correspond to standard database commands and operators (Table 3), we formulate them as SQL queries over article metadata, enabling automatic derivation of ground-truth answers through query execution without manual annotation. We then exploit LLM-based SQL-to-text generation (Li et al., 2023; Hui et al., 2024) to convert queries into natural-language questions, and ensure that their meaning is preserved through round-trip verification (SQL $\rightarrow$ question $\rightarrow$ SQL).

As an example, consider the first question in Table 1: *"What is the highest number of authors that any single article has?"* Although the question seems simple, answering it requires locating the author list of every article in the context, counting the authors per article, comparing these counts, and identifying the maximum. The numerical operations in SciTrek underpin complex questions in expert-curated benchmarks (Wolfson et al., 2020; 2025) and recur in human-written questions in OpenScholar (Asai et al., 2024), LongBench v2 (Bai et al., 2025), and Loong (Wang et al., 2024c) (Appendix A).

Our questions are grounded in surface-level metadata rather than scientific content, and some may not resemble queries a human would naturally pose. Nevertheless, they probe prerequisite capabilities: a model that cannot reason accurately about such easily identifiable elements is unlikely to succeed on deeper analytical tasks. Indeed, when models are given the metadata as database tables instead of full-text articles, accuracy rises sharply (Section 4), indicating that the primary challenge lies in extracting and tracking relevant information across long contexts rather than in the numerical operations themselves. SciTrek is diagnostic in two

Dataset	N-Ctx	Numerical	Scalable	Length	Use	Example Question
NeedleBench	✗	✗	✓	128K	E	What legendary item is hidden on Emerald Island?
Ada-LEval	✗	✗	✓	128K	E	What is the correct order of the segments?
BABILong	✗	✗	✓	10M	E	Where is Mary?
HELMET	✗	✗	✗	128K	E	Who set the fire in one tree hill?
LIFBENCH	✗	✗	✓	128K	E	Retrieve the entry at position 8th in the list.
RULER	✗	✗	✓	128K	E	Find all variables that are assigned the value 12345.
OpenScholar	✓$^{S/M}$	✗	✗	3K*	E	Compile a list of reviews [...], and identify the most promising [...]
LongBench v2	✓$^{S/M}$	✗	✗	128K	E	How long have I been living in my current apartment in Shinjuku?
LongMemEval	✓^{M}	✗	✗	2M	E	How many bikes do I currently own?
L-Eval	✓$^{S/M}$	✗	✗	200K	E	How do I know when I should apply for Medicare?
HoloBench	✗	✓	✓	64K	E	What are the names of wines and their corresponding grape types?
MathHay	✗	✓	✓	128K	E	What is the total number of points scored by LeBron [...] combined?
Loong	✓^{M}	✓	✗	250K	E	Which company has the highest non-current assets?
LoongRL	✗	✗	✓	16K	T	When did Louis Deniset's political party form?
LongRLVR	✓^{S}	✗	✓	64K	T	What is the name of the Bastard's father, what honor does he receive [...] to England?
SciTrek (ours)	✓^{M}	✓	✓	1M	E, T	How many articles are cited by other articles but do not cite any other articles?

Table 2: Representative datasets for evaluating and post-training long-context language models. **N-Ctx**: whether the dataset uses naturally occurring input contexts; for natural contexts, we further indicate whether answering the questions requires a single (^{S}) or multiple (^{M}) documents. **Numerical**: whether the dataset requires numerical reasoning over long contexts; **Scalable**: whether the dataset can be extended with minimal human effort (e.g., to longer contexts or larger datasets); **Length**: maximum supported input length in tokens; *: length is based on the retrieved text rather than the full articles which are not available. **Use**: whether the dataset is intended for training (T), evaluation (E), or both.

complementary ways: the table-article gap isolates the cost of reasoning over unstructured long contexts, and the underlying executable SQL queries provide an interpretable reference for fine-grained error analysis.

While our automated pipeline can generate an unlimited number of question-answer pairs, SciTrek provides a curated test set of 2,121 and a training set of 19,543 instances over scientific articles, spanning context lengths of 64K, 128K, 512K, and 1M tokens. We use the test set to conduct an extensive, fine-grained evaluation of frontier LLMs with context windows exceeding 128K tokens. The training set, in turn, is used to post-train open-weight models, improving their numerical reasoning on long-context tasks in other domains, including LongBench v2 (Bai et al., 2025) and Loong (Wang et al., 2024c). Our contributions are as follows:

- We develop a long-context numerical reasoning dataset grounded in scientific articles. The underlying SQL queries for each question enable fine-grained diagnosis of model failures, and the automated data curation pipeline scales effortlessly to longer contexts and larger datasets with minimal human effort.
- Extensive evaluation reveals that frontier LLMs struggle even with the ostensibly simple reasoning questions in SciTrek, and that performance declines sharply as context length grows. Fine-grained analysis traces these failures to citation-related questions, sorting, and compound logical conditions, particularly those involving negation.
- Beyond its diagnostic value, SciTrek serves as an effective post-training corpus: fine-tuning open-weight LLMs on its training set improves their numerical reasoning, with gains that generalise to out-of-domain long-context tasks.

2 Related Work

Retrieval-focused Long-context Datasets The increasing interest in long-context language models has driven the development of numerous datasets, mostly targeting benchmarking. Initial efforts primarily evaluate whether models can accurately retrieve specific information in long contexts, typically through Needle-In-A-Haystack tasks such as NeedleBench (Li et al., 2024). Subsequent benchmarks extend this paradigm by adding distractor information or scattering relevant content across the input, including AdaL-Eval (Wang et al., 2024a), HELMET (Yen et al., 2025), LIFBENCH (Wu et al., 2025b), LongBench (Bai et al., 2024), and RULER (Hsieh et al., 2024). While these datasets usefully stress-test information retrieval, they do not assess whether models can reason over the retrieved information.

Reasoning over Synthetic Contexts A second line of work targets reasoning over long contexts but relies on artificial inputs. BABILong (Kuratov et al., 2024) embeds short reasoning tasks in long streams of unrelated text, HoloBench (Maekawa et al., 2025) considers numerical reasoning over various types of tables, and MathHay (Wang et al., 2024b) focuses on mathematical reasoning within long documents. However, they all construct their contexts synthetically — by injecting noise into short passages or generating inputs with LLMs — so models may exploit distributional regularities of synthetic text that do not transfer to naturally occurring documents.

Reasoning over Natural Contexts Several datasets combine reasoning with natural contexts, including LongBench v2 (Bai et al., 2025), LongMemEval (Wu et al., 2025a), and LEval (An et al., 2024); Loong (Wang et al., 2024c) and OpenScholar (Asai et al., 2024) evaluate reasoning over scholarly documents in particular. However, these benchmarks rely on expert-curated questions and answers, which caps them at evaluation scale; LongBench v2, for instance, comprises 503 questions in total, far short of the quantities required for post-training, and manual curation cannot keep pace with rapidly expanding context windows, increasing the risk of data contamination and obsolescence. Moreover, because the reasoning required to reach a correct answer is typically not made explicit, fine-grained failure analysis remains difficult.

Synthetic Data as Post-training Corpora Most long-context datasets grounded in natural inputs focus on evaluation (Wang et al., 2024c; Wu et al., 2025a; Bai et al., 2025). For post-training, LoongRL (Wang et al., 2026), LongRLVR (Chen et al., 2026), and QwenLong1.5 (Shen et al., 2025) develop datasets to improve long-context reasoning, but all rely on synthetic contexts, and none explicitly targets numerical reasoning over long inputs (see the discussion in Appendix A). Numerical reasoning itself has been studied extensively in short contexts, most prominently in DROP (Dua et al., 2019); SciTrek extends this line of work to inputs several orders of magnitude longer. Together, this leaves a scarcity of public training data for numerical reasoning over long, natural contexts.

Unlike existing datasets, SciTrek is, to our knowledge, the first long-context reasoning dataset that is simultaneously grounded in naturally occurring documents and automatically scalable. Its contexts are full-text scientific articles, and its numerical-reasoning questions are generated automatically from article metadata (titles, authors, and references) by repurposing resources and models from text-to-SQL research (Yu et al., 2018; Li et al., 2023; Hui et al., 2024) for SQL-to-question generation. Loong is closest to SciTrek in that it targets numerical reasoning over natural legal, financial, and scientific documents. However, its reliance on substantial manual annotation prevents automatic scaling, and its evaluation is limited to contexts of up to 250K tokens. A detailed comparison between SciTrek and existing datasets is presented in Table 2.

3 The SciTrek Dataset

Figure 1 illustrates the SciTrek curation pipeline: we first assemble collections of scientific articles as contexts at varying lengths (i.e., 64K, 128K, 512K, and 1M), then construct databases representing article metadata and generate SQL queries with answers obtained via database execution. Finally, we convert the SQL queries into natural language questions. We describe each stage in detail below.

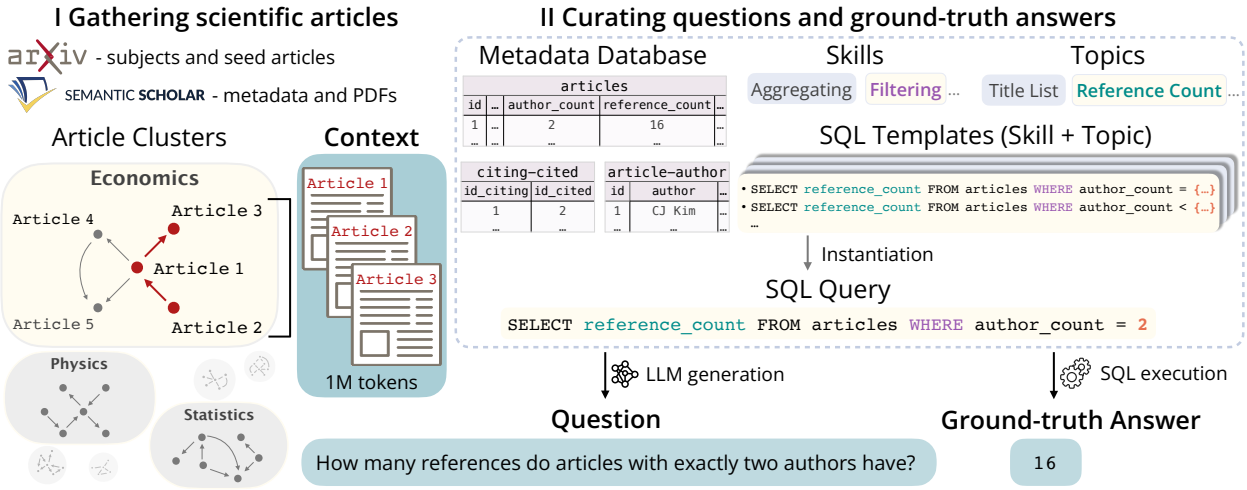

Figure 1: Overview of SciTrek construction pipeline. We first gather scientific article collections of varying scales; then generate SQL queries and their answers from databases storing collection-specific metadata; and finally convert the SQL queries to natural language questions. The final dataset comprises three components highlighted in blue: full-text scientific articles as the input context, the natural language question, and the corresponding answer.

3.1 Gathering Scientific Article Collections

SciTrek is constructed from scientific articles sourced from Semantic Scholar.[3] To ensure the representation of diverse topics, we obtain an initial set of seed articles from eight subjects: Computer Science (CS), Economics, Electronic Engineering (EE), Math, Physics, Biology, Finance, and Statistics.[4] For each subject, we select two seed articles with more than 100 citations since 2020 and for each article, we retrieve related articles via the Semantic Scholar API.[5] To ensure broad coverage, we include two-hop related articles identified via Semantic Scholar's citation graph based on the reference and citation list. For each seed article, we randomly sample ten first-hop neighbors, and for each of these we further sample five second-hop neighbors forming an article cluster. Since full texts are required, we filter out articles without PDFs. This process yields 16 article clusters comprising 662 scientific articles with PDFs across eight subjects (see step I in Figure 1). Finally, we convert the collected PDFs to Markdown using Marker.[6]

From these clusters, we create article collections at varying context lengths for our question-answering task. By concatenating the Markdown texts, we produce collections of 64K, 128K, 512K, and 1M tokens.[7] Each collection is initialised with a randomly selected article, and further articles are appended until the target length is reached. We employ two expansion strategies: (1) random sampling from the clusters, and (2) traversal of the citation graph within each cluster (using both depth-first and breadth-first search) to construct collections that preserve citation relations among articles. Random groupings expose models to heterogeneous, cross-domain contexts, while citation-based groupings introduce richer relational structure that demands more sophisticated long-context reasoning. Each collection contains at least four articles, and no two collections share more than half of their articles. From the 662 scientific articles across 16 clusters, we construct 2,612 collections spanning all four length levels: 2,027 via random sampling and 585 via graph traversal.

[3] https://www.semanticscholar.org/

[4] Following the typology of subjects from https://arxiv.org/.

[5] Semantic Scholar provides APIs to retrieve scientific articles together with metadata including titles, authors, a reference list with articles that the current article cites, and a citation list with articles that cite the current article. Throughout this paper, a reference is a bibliographic entry listed in the reference section of a scientific article, whereas a citation is an in-text acknowledgement of another's work within the article.

[6] https://github.com/datalab-to/marker

[7] We can easily construct collections with arbitrary lengths exceeding 1M tokens.

Skill	Count	Example Query Template
Aggregating	20	`SELECT MAX(author_count) FROM articles`
Sorting	27	`SELECT title_word_count FROM articles ORDER BY author_count ASC`
Filtering	107	`SELECT author_name FROM article_author WHERE author_position = {author-position}`
Filtering+ Aggregating	107	`SELECT SUM(title_word_count) FROM articles WHERE reference_count = {reference-count}`
Filtering+ Sorting	106	`SELECT author_count FROM articles WHERE title_word_count % 2 = 1 ORDER BY title_word_count DESC`
Relational Filtering	20	`SELECT COUNT(*) FROM articles WHERE article_id NOT IN (SELECT article_id_citing FROM citing_cited) AND article_id IN (SELECT article_id_cited FROM citing_cited)`

Table 4: SQL templates representing different information processing skills. {author-position}, {reference-count} are placeholders.

3.2 Creating Databases and SQL Queries

Once the article collections are assembled, we construct a database for each (see step II in Figure 1), focusing on a core subset of elements shared across all articles: titles, authors, and references. Based on these key elements, each database comprises three tables: *articles*, *article-author*, and *citing-cited*. The *articles* table contains metadata such as the title, reference count, and title word count. The *article-author* table captures author information, including names and their positions in the corresponding author list. The *citing-cited* table records citation relations among articles. A detailed description of the database tables is provided in Appendix B. This information is obtained from Semantic Scholar or derived via simple preprocessing, such as splitting titles and counting words. Manual corrections were applied as needed to align with the full-text Markdown articles.

The core SQL[8] commands summarized in Table 3 form the foundation for building SQL queries, often combined with operators (Comparison, Arithmetic, Logical) to create more complex queries (since `WHERE` is used to filter data, it always works in conjunction with SQL operators). Aside from the SQL commands in Table 3 (Aggregating, Sorting, and Filtering), we define composite commands based on their combinations (i.e., Filtering+Aggregating and Filtering+Sorting). Using these, we manually create SQL query templates targeting different topics related to key elements of scientific articles, including Author Count, Author List, Reference Count, Title List, and Title Word Count. This process is illustrated in Figure 1 (right panel) and example templates are shown in Table 4; Note that some templates include placeholders to be instantiated with specific values. Finally, to capture authorship and citation relations, we introduce Relational Filtering and the topics of Author Relation and Citation Relation, which specifically target authorship and citation relations.

SQL Commands	
Aggregating	`MAX`, `MIN`, `SUM`, `AVG`, `COUNT`, `DISTINCT`
Sorting	`ORDER BY`, `ASC`, `DESC`, `GROUP BY`
Filtering	`WHERE`

SQL Operators	
Comparison	=, >, <, >=, <=, <>, `LIKE`
Arithmetic	+, −, ∗, /, %
Logical	`AND`, `NOT`, `OR`, `BETWEEN`, `IN`

Table 3: Core commands and operators in SQL.

We collectively refer to the SQL commands in Table 4 as information processing skills, since they test different information processing capabilities. We have various templates per skill, designed to be applicable across collections. For each collection, we randomly select 10 templates and instantiate them with collection-specific values for all placeholders. We then execute these queries against the corresponding database to generate ground-truth answers.

[8]Structured Query Language (SQL) is a standardized language for managing and querying relational databases.

Context Length	#Samples	#Context Articles	Question Length	Gold-answer Length
64K	112	4.2	14.6	6.2
128K	728	5.7	16.9	7.9
512K	667	22.7	18.1	30.9
1M	614	46.3	18.6	63.6

Table 5: Descriptive statistics for the SciTrek test set. **Context Length**: target context length in tokens; **#Samples**: number of question–answer pairs; **#Context Articles**: mean article count per context; **Question Length** and **Gold-answer Length**: mean length for questions and ground-truth answers in words.

3.3 Converting SQL Queries to Natural Language

We use Qwen2.5-Coder-32B-Instruct (Hui et al., 2024) to convert SQL queries into natural language questions. To ensure queries and questions are meaning-preserving, we validate each generated question by converting it back to SQL and verifying that both queries produce identical results when executed against the collection database. For each SQL-collection pair, we repeat this process up to 10 times; if no valid question is obtained, we discard the query for that collection. Using the prompts in Appendix F, we successfully generate natural language questions for 82.9% of SQL-collection pairs. Following this, we obtain 2,121 test questions for evaluating models against the four context lengths defined in Section 3.1, and 19,543 instances for training (see post-training experiments in Section 5). We split the data at the QA-pair level, ensuring that the QA pairs in the training, validation, and test sets are distinct, although source articles and SQL templates may overlap across splits. Table 5 presents descriptive statistics for the SciTrek test set. For each context length, our test partition covers all information processing skills and question topics described in Section 3.2. The distribution of SQL commands and operators is provided in Appendix B, Table 14.

3.4 Data Quality Validation

To validate SciTrek's quality, we crowdsourced human annotations through Prolific.[9] Annotators (English native speakers from the US or the UK) were asked to answer natural language questions from a random sample of 120 instances, with 20 instances on average representing each of the six skills listed in Table 4.[10] Crowdworkers answered questions using database tables rather than full-text articles, as documents spanning 1M tokens are impractical for humans to review. Each instance was annotated by three participants following the same instructions as those used for model testing with database tables as context (Figure 8, Appendix C). In each annotation session, crowdworkers were presented with three questions over different contexts. To ensure annotation quality, each session included a quality control question. On average, annotators spent about 5.5 minutes per session and were compensated above the UK living wage, at £12 per hour.

We measured inter-annotator agreement and the alignment between database-derived answers (from SQL query execution) and human responses using exact match. Annotators were considered in agreement if two or more provided the same answer. Similarly, database answers were considered aligned with human responses if they matched the responses of two or more annotators. Table 6 reports average agreement and alignment across 120 instances, showing strong inter-annotator agreement and high consistency between database-derived and human answers.

Skill	Agree (%)	Align (%)
Aggregating	85.7	85.7
Sorting	85.0	80.0
Filtering	85.7	85.7
Filtering+Aggregating	95.0	85.0
Filtering+Sorting	89.5	89.5
Relational Filtering	89.5	73.7
All	88.3	83.3

Table 6: Inter-annotator agreement on 120 sampled answers. Answers obtained from SQL execution are considered aligned if they match those provided by two or more annotators.

Manual inspection of all 120 questions given to annotators revealed that 3 of the 19 Relational Filtering questions were ambiguous, explaining the gap between alignment and agreement for this category in Ta-

[9] https://www.prolific.com

[10] Aggregating: 21, Sorting: 20, Filtering: 19, Filtering+Aggregating: 20, Filtering+Sorting: 21, and Relational Filtering: 19.

			Full-text Articles				Database Tables			
Model	**Ctx**	**Think**	**F-64**	**F-128**	**F-512**	**F-1024**	**D-64**	**D-128**	**D-512**	**D-1024**
Qwen-2.5-Coder + SQL execution	—	—	38.4	35.3	34.1	25.6	90.6	89.4	86.8	87.5
Qwen2.5-7B-Instruct-1M	1M	✗	4.5	2.8	0.3	0.0	20.5	14.3	5.0	2.0
Qwen2.5-14B-Instruct-1M	1M	✗	8.3	6.5	1.6	0.1	33.3	27.2	11.0	5.9
Qwen3-4B-Instruct-2507	256K	✗	2.1	7.2	—	—	25.9	16.9	6.9	2.8
Qwen3-4B-Thinking-2507	256K	✓	41.1	29.3	—	—	90.2	83.1	71.8	52.3
Qwen3-30B-A3B-Instruct-2507	256K	✗	5.4	3.2	—	—	29.8	21.2	6.0	4.2
Qwen3-30B-A3B-Thinking-2507	256K	✓	53.3	42.0	—	—	92.3	86.2	73.5	61.1
Gemma-3-27B-IT	128K	✗	6.2	3.4	—	—	31.8	25.0	11.3	6.1
Llama-4-Scout-17Bx16E-Instruct	10M	✗	5.4	2.8	1.3	1.1	28.6	19.0	7.4	4.0
Llama-3.3-70B-Instruct	128K	✗	8.3	3.5	—	—	47.0	36.2	14.6	8.1
DeepSeek-R1-Distill-Llama-70B	128K	✓	22.0	6.0	—	—	83.3	74.2	56.8	42.2
GPT-4.1	1M	✗	21.1	11.7	3.9	2.5	69.3	53.8	24.8	16.1
Gemini 2.5 Pro	1M	✓	41.7	26.0	⋆	⋆	91.7	83.5	55.4	31.5
o4-mini	195K	✓	61.0	46.5	—	—	95.2	87.8	79.4	72.6

Table 7: Exact match (%) on SciTrek for the SQL execution-based method (top), open-weight (middle), and proprietary (bottom) models, under two context settings: full-text scientific articles and the corresponding database tables. **Ctx** is the maximum input length each model supports; **Think** (✓, shaded rows) marks models prompted with chain-of-thought, while non-thinking models (✗) answer directly. F-* denotes test instances using full-text article collections of up to 64K, 128K, 512K, and 1M tokens. D-* denotes test data instances using the database tables corresponding to F-64, 128, 512, and 1024. For any given article collection, the database representation is substantially shorter (mean ∼2K tokens), and can therefore be passed to all models regardless of their context limit. — : the model cannot process the given context size; ⋆: not evaluated due to prohibitive computational cost.

ble 6. While Qwen2.5-Coder-32B-Instruct generally converts SQL queries to natural language accurately, the complexity of Relational Filtering queries can introduce ambiguity.

4 Frontier-Model Evaluation

We first use SciTrek to evaluate both proprietary and open-weight models from various families that support contexts exceeding 128K and have shown strong performance on language understanding and mathematical reasoning benchmarks such as MMLU (Hendrycks et al., 2021a) and MATH (Hendrycks et al., 2021b).

For proprietary models, we include o4-mini[11] and Gemini 2.5 Pro (Gemini Team, 2025), which are widely regarded as the leading models for long-context reasoning. For open-weight models, we select the largest model we can feasibly run with our resources from each model family.[12] For example, we use Llama-4-Scout-17Bx16E-Instruct instead of Llama-4-Maverick-17B-128E-Instruct (Llama Team, 2025), and the distilled variant of DeepSeek-R1 (DeepSeek-AI et al., 2024). Table 7 provides an overview of the models we consider, which vary in parameter scale and supported context length (most were released within the past year). For detailed model descriptions and settings, refer to Appendix C. We evaluate models in two context settings: (1) using the full-text scientific articles within a given collection as context, and (2) using only the corresponding database tables. We employ chain-of-thought prompting for thinking models, allowing them to output reasoning steps before the answer, while prompting non-thinking models to output the answer without any intermediate reasoning steps (actual prompts for these models are included in Appendix C). We assess performance using average exact match and F1, as the expected outputs are factual items with minimal variation, such as specific numbers, author names, or article titles. For higher confidence, reported metrics are averages across three runs.

[11] https://developers.openai.com/api/docs/models/o4-mini
[12] Experiments were conducted on 4 NVIDIA HGX H200s.

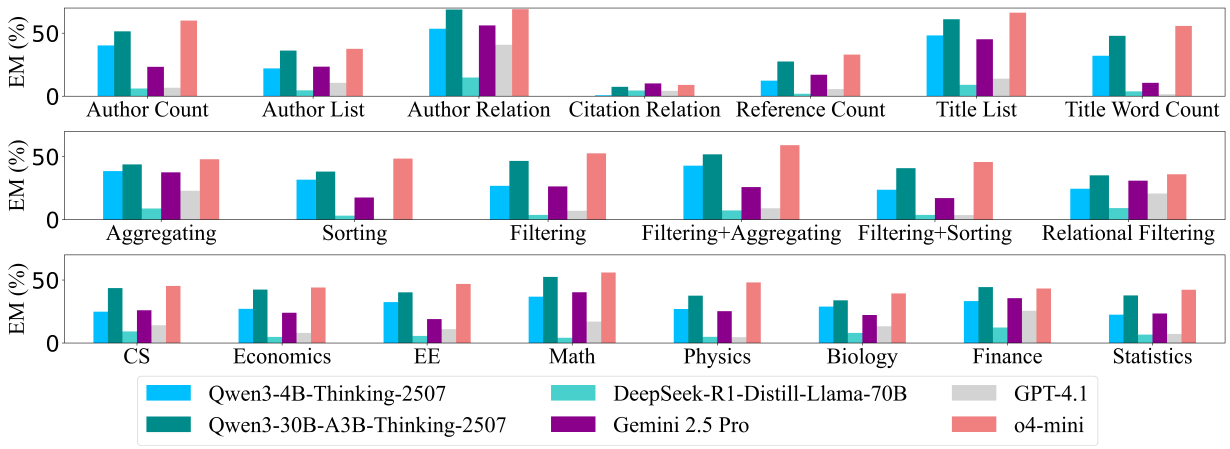

Figure 2: Fine-grained analysis for best performing models in terms of exact match (EM). We examine how model performance varies across question topics (top), information processing skills (middle) and subjects (bottom) when using full-text scientific articles as context at an input length of 128K tokens.

Model performance is summarised in Table 7. SciTrek proves challenging across all evaluated models, especially with full-text articles as context: performance degrades for every model as inputs grow longer and task size gets larger, and the same trend holds, more mildly, with database tables. The single largest factor is thinking mode rather than scale or provenance: at matched size, thinking variants outperform their non-thinking counterparts by 30–50 EM points (e.g., Qwen3-30B-A3B: 5.4 vs. 53.3 at 64K), the best open-weight thinking model surpasses Gemini 2.5 Pro, and o4-mini is strongest overall. Notably, non-thinking models struggle even on the ∼2K-token database tables (20.5–47.0 EM at 64K), indicating that a substantial part of the difficulty lies in executing multi-step numerical operations, in addition to locating and tracking information in long unstructured inputs.

We also evaluate an SQL execution-based baseline (first row of Table 7): Qwen2.5-Coder-32B-Instruct converts each natural-language question into an SQL query, which is executed against the corresponding database. In the database-table setting, the query runs on the gold tables; in the full-text setting, we first construct the tables from titles, authors, and references extracted from the articles by the same model.[13] Execution on *gold tables* approaches ceiling (≥86.8 EM) and constitutes an upper bound for this extract-then-execute approach, whereas execution on *extracted tables* performs far worse (25.6–38.4 EM), showing that the main bottleneck in this pipeline is extracting accurate metadata from long unstructured text, rather than generating and executing the SQL query. Similar trends hold under F1 (Appendix D).

Model	#Articles	Q-Len	SQL-Len	GA-Len
Qwen2.5-7B-1M	−0.04	−0.15*	0.11	−0.01
Qwen2.5-14B-1M	−0.04	−0.16*	0.07	−0.06
Qwen3-4B-Instruct-2507	−0.01	−0.16*	0.05	−0.03
Qwen3-4B-Thinking-2507	−0.06	−0.14*	−0.19*	−0.01
Qwen3-30B-A3B-Instruct-2507	−0.02	−0.15*	0.06	0.08
Qwen3-30B-A3B-Think-2507	−0.12*	−0.08	−0.19*	−0.02
Gemma-3-27B-IT	−0.08	−0.14*	0.02	−0.08
Llama-4-Scout	−0.10*	−0.17*	−0.03	−0.03
Llama-3.3-70B	−0.06	−0.16*	−0.03	−0.05
DeepSeek-R1-Distill-Llama-70B	−0.15*	−0.16*	−0.02	−0.08
Gemini 2.5 Pro	−0.16*	−0.22*	−0.03	−0.01
GPT-4.1	−0.07	−0.21*	0.06	0.06
o4-mini	−0.14*	−0.09	−0.25*	−0.06

Table 8: Pearson correlation between per-instance exact match and four properties of each instance (with full-text articles as context, 128K tokens). **Q-Len:** question length, **SQL-Len:** SQL query length, and **GA-Len:** gold answer length; all lengths are counted in space-separated words. *: $p < 0.05$.

We further conduct a fine-grained analysis for the best six models across question topics (e.g., author count, reference count, citation relations), skills (e.g., aggregating, sorting), and subjects (e.g., Economics, Biology)

[13]Extraction prompts are given in Appendix F.

Model	Post-training	Aggregating	Sorting	Filtering	F+A	F+S	Relational	Overall
Qwen3-235B-A22B-Instruct	Zero-shot	48.0	47.9	43.1	54.8	40.6	36.8	42.9
Qwen2.5-7B-Instruct-1M	Zero-shot	7.8	0.0	1.2	2.6	0.0	6.1	3.3
	SFT	14.7	0.0	4.2	13.4	2.4	30.4	15.4
	GRPO	34.3	12.5	22.9	33.1	15.0	24.5	23.8
Qwen3-4B-Instruct	Zero-shot	5.9	0.0	0.5	1.3	0.0	5.2	2.5
	SFT	17.6	0.7	7.9	18.6	4.4	36.6	19.8
	GRPO	43.1	21.5	31.5	47.2	25.8	30.4	32.9
Gemma3-4B-IT	Zero-shot	3.9	0.0	0.0	1.7	0.0	1.3	1.0
	SFT	3.9	0.0	3.5	12.6	3.1	21.8	11.6
	GRPO	7.8	0.0	2.1	7.8	0.2	10.1	5.9

Table 9: In-domain results of different numerical reasoning skills on the SciTrek test set (contexts≤128K); models are evaluated using exact match (%) in a zero-shot setting or after post-training on SciTrek with SFT or GRPO. We additionally report zero-shot results for Qwen3-235B-A22B-Instruct as a reference point at larger scale. **F+A**: Filtering + Aggregation; **F+S**: Filtering + Sorting; **Relational**: Relational Filtering.

when using full-text articles as context (shown in Figure 2). Model performance shows little variation by subject, which indicates that domain-specific content does not have an effect on our evaluation. However, most models struggle more with sorting tasks, while performing better on aggregation. Performance is lowest on citation-related questions (i.e., Citation Relation, Reference Count), and somewhat higher on author- and title-related questions. This is possibly because larger reasoning elements make it more challenging to get the answer. To understand which factors predict failure, Table 8 reports correlations between per-instance exact match and four properties of each instance: the number of input articles, and the lengths (in words) of the question, the underlying SQL query, and the gold answer. All correlations are weak ($|r| \leq 0.25$), but a consistent pattern emerges. For non-thinking models, performance correlates significantly only with question length, suggesting sensitivity to surface complexity. For thinking models (Qwen3-Thinking variants, o4-mini), the dominant correlate shifts to the length of the underlying SQL query (a proxy for the structural complexity of the required computation) while question length loses significance for the strongest of them. Gold-answer length correlates with performance for no model, indicating that failures are not driven simply by questions demanding longer outputs.

Finally, to characterise *how* models fail, we manually analysed 60 randomly selected instances at 128K input length on which all zero-shot (and post-trained) models fail (10 per skill in Table 4), and identified four recurring failure patterns (further details in Appendix E): (1) **Fallback to NULL.** Many open-weight models frequently output "NULL", the answer our prompt reserves for cases where no answer can be found in the context, particularly on filtering tasks, suggesting they resort to this fallback rather than reasoning over the provided input. (2) **Format non-compliance.** Models often violate the specified answer format, especially on sorting questions, for instance returning author lists when sorted author counts are requested, or lists when scalar aggregates are expected; GPT-4.1 does so far more often than any other model. (3) **Negation.** All models struggle with questions involving negation (e.g., "not", "never"). (4) **Hallucination of specific facts.** As contexts grow, thinking models (Qwen3-4B-Thinking-2507, Qwen3-30B-A3B-Thinking-2507, DeepSeek-R1-Distill-Llama-70B) produce plausible high-level reasoning plans yet hallucinate the specific facts needed to execute them, particularly counts, such as the number of reference items.

5 Post-training Experiments

Using the automated pipeline from Section 3, we generate a training set of 19,543 instances spanning the four context lengths (64K, 128K, 512K, and 1M tokens). Because these instances exercise diverse numerical operations over long, natural contexts without depending on domain-specific knowledge, they are well suited to post-training open-weight LLMs for long-context numerical reasoning. We evaluate both in-domain gains on SciTrek and transfer to out-of-domain long-context benchmarks.

Model	Post-training	LongBench v2	Loong-Academic	Loong-Financial
Qwen3-235B-A22B-Instruct	Zero-shot	50.8	49.2	53.3
Qwen2.5-7B-Instruct-1M	Zero-shot	33.3	31.3	38.5
	SFT	34.4 (+1.1)	39.2 (+7.9)	39.5 (+1.0)
	GRPO	35.7 (+2.4)	33.5 (+2.2)	42.0 (+3.5)
Qwen3-4B-Instruct	Zero-shot	35.4	24.9	37.8
	SFT	13.3 (−22.1)	46.4 (+21.5)	38.8 (+1.0)
	GRPO	45.1 (+9.7)	34.3 (+9.4)	40.1 (+2.3)
Gemma3-4B-IT	Zero-shot	26.8	3.6	25.9
	SFT	1.0 (−25.8)	11.8 (+8.2)	18.0 (−7.9)
	GRPO	23.0 (−3.8)	4.5 (+0.9)	25.2 (−0.7)

Table 10: Out-of-domain transfer results on LongBench v2 and Loong for models evaluated zero-shot or after post-training on SciTrek with SFT or GRPO. Performance on LongBench v2 is multiple-choice accuracy (%); performance on Loong is answer quality on a 1–100 scale judged by GPT-5 mini; higher is better in both cases; We follow the evaluation protocols of Bai et al. (2025) and Wang et al. (2024c). Parenthesised values give the change relative to the same model's zero-shot performance (teal: improvement; gray: degradation). Qwen3-235B-A22B-Instruct is included as a zero-shot reference for scale.

We experiment with two well-established post-training techniques: supervised fine-tuning (SFT) and reinforcement learning with verifiable rewards (RLVR).[14] Owing to the high computational cost, we train on contexts of up to 128K tokens (7,703 instances) and report results for three models from different families: Qwen2.5-7B-Instruct-1M, Qwen3-4B-Instruct, and Gemma3-4B-IT.[15] For SFT, we train for 3 epochs with a batch size of 32, a learning rate of 2×10^{-6}, and a warm-up ratio of 0.05. For RL, we use GRPO (Shao et al., 2024) with a summed EM and F1 reward, which we found to improve accuracy over time while still providing signal on difficult questions; GRPO has proven effective on similar verifiable long-context tasks (Shao et al., 2024; Mroueh, 2025; Gurung & Lapata, 2025). Owing to the computational cost of GRPO optimisation, we restrict our training to a single epoch (approximately five days) with a maximum generation budget of 2,048 tokens. All results are averaged over three responses per question to reduce sampling variance.

In-domain Results We first investigate whether post-training on SciTrek improves performance on SciTrek itself, and which numerical reasoning skills are more difficult to acquire. Table 9 compares model performance after post-training on SciTrek with the zero-shot setting. Both SFT and GRPO substantially improve performance, with GRPO yielding larger performance gains, particularly for Qwen3-4B-Instruct (comparable to Qwen3-235B-A22B-Instruct on aggregating-related tasks), though the smaller models generally remain well below the performance of Qwen3-235B-A22B-Instruct. Across different numerical reasoning skills, sorting-related tasks are the most difficult to learn, whereas aggregation-related tasks benefit the most from post-training. This suggests sorting demands a qualitatively different capability that the two standard post-training techniques fail to unlock.

Out-of-domain Transfer We next evaluate whet her these gains extend beyond SciTrek altogether, using two long-context benchmarks from different domains that nonetheless require the numerical reasoning skills SciTrek targets: Loong (Wang et al., 2024c) and LongBench v2 (Bai et al., 2025). Loong is designed to evaluate long-context language models through extended multi-document question answering with free-form answers, in which every document in each test case must be considered to derive the final answer. It spans four task types (Spotlight Locating, Comparison, Clustering, and Chain of Reasoning) across three domains (Financial Reports, Academic Papers, and Legal Cases) with context lengths ranging from 10K to beyond 200K tokens. We evaluate on the English-language samples with context lengths up to 128K tokens, without any further adaptation, using 309 instances from Financial Reports and 119 from Academic Papers.[16]

[14] SFT trains models to map the input directly to the answer, whereas RLVR encourages models to generate reasoning traces before answering.

[15] Due to computational resource constraints, our post-training experiments are confined to models under 10 billion parameters. However, since our data curation pipeline is agnostic to model architecture and scale, we expect the benefits to generalise to larger models as well.

[16] The data instances from the Legal Cases domain are omitted, as they are entirely in Chinese.

Post-training Corpus	#Samples	LongBench v2	Loong-Academic	Loong-Financial	SciTrek
LoongRL	21,024	36.1	32.4	33.8	25.4
LongRLVR	46,000	36.8	**36.2**	34.2	28.5
SciTrek	7,703	**45.1**	34.3	**40.1**	**32.9**

Table 11: Qwen3-4B-Instruct is post-trained with GRPO (one epoch) on each corpus and evaluated on LongBench v2 (multiple-choice accuracy, %), Loong (answer quality on a 1–100 scale judged by GPT-5 mini; English subsets, contexts ≤128K), and SciTrek test set (exact match, %; contexts ≤128K); higher is better. #Samples: training instances per corpus.

Unlike Loong, LongBench v2 is designed to evaluate long-context language models on deep understanding and reasoning across multiple real-world tasks with document, dialogue and code data. It consists of 503 challenging multiple-choice questions, with contexts ranging from 8K to 2M words. We evaluate on samples with context lengths up to 128K tokens. We employ single-choice question accuracy (%) and LLM-based answer quality (scored on a 1–100 scale by GPT-5 mini as a judge) as the evaluation metrics, following the evaluation protocols of Bai et al. (2025) and Wang et al. (2024c), respectively.

Results in Table 10 show that post-training on SciTrek generally improves out-of-domain performance, but the two methods behave very differently. GRPO yields consistent gains for the Qwen2.5-7B and Qwen3-4B models across all three benchmarks (up to +9.7 for Qwen3-4B on LongBench v2), bringing a 4B model within reach of the much larger Qwen3-235B zero-shot reference on LongBench v2. SFT is far less stable: while producing the largest single gain (+21.5 for Qwen3-4B on Loong–Academic), it severely degrades LongBench v2 performance for Qwen3-4B and harms Gemma3-4B on two of three benchmarks. We attribute this asymmetry to output format: SciTrek and Loong both require free-form answers, whereas LongBench v2 is multiple-choice, and SFT's stronger imprinting of SciTrek's terse answer format appears to interfere with the multiple-choice setting, while GRPO, which optimises rewards over the model's own outputs, transfers across both formats. Gains are also uneven across base models: Gemma3-4B, the weakest zero-shot model, benefits little from GRPO (within ±4 points on all three benchmarks) and suffers the largest SFT degradation (down to 1.0 on LongBench v2), suggesting that transfer from SciTrek depends on sufficient base-model capability.

Comparison with Existing Post-training Corpora Finally, we compare SciTrek against two recent post-training corpora for long-context reasoning: LoongRL (Wang et al., 2026), which expands Wikipedia-grounded short-context QA into 16K-token inputs by inserting distractor documents and multi-hop key-value chains that models must trace to recover the question, and LongRLVR (Chen et al., 2026), which pairs natural documents of 8K–64K tokens (books, arXiv papers, code) with questions and answers synthesised by Qwen3-235B-A22B-Instruct.[17] For a controlled comparison, we post-train the same model (Qwen3-4B-Instruct) with the same method (GRPO, one epoch, 2,048-token generation budget) on each corpus. As shown in Table 11, training on SciTrek gives the best result on LongBench v2 and Loong–Financial, and remains competitive on Loong–Academic (34.3 vs. LongRLVR's 36.2), despite using roughly a third of the training samples. As expected, it also attains the highest in-domain SciTrek accuracy. We attribute this efficiency primarily to data design. SciTrek grounds reasoning in naturally occurring documents and requires verifiable numerical operations over consistently identifiable elements, providing a dense, checkable reward signal. LoongRL's artificially assembled contexts, by contrast, likely reward puzzle-specific tracing skills that transfer less readily to natural documents, while LongRLVR's model-synthesised questions may inherit the generator's distributional biases.

6 Conclusion

We introduced SciTrek, a synthetic dataset for evaluating and improving long-context numerical reasoning in LLMs, whose questions and ground-truth answers are generated automatically as SQL queries over metadata of scientific articles (titles, authors, and references). Although the questions require only the ability to locate and manipulate easily identifiable elements, our evaluation of thirteen open-weight and proprietary LLMs

[17] QwenLong-L1.5 (Shen et al., 2025) is excluded because its dataset was not publicly available at the time of writing.

shows they pose a substantial challenge: the best model reaches 46.5% exact match at 128K tokens, performance degrades sharply with context length, and visible step-by-step reasoning (thinking mode) matters far more than scale or provenance. Fine-grained analysis exposes systematic failure modes, most notably compound logical conditions involving negation and, for thinking models, hallucination of the specific facts (particularly counts) needed to execute otherwise sound reasoning plans. Post-training on SciTrek yields substantial in-domain gains and transfers to out-of-domain long-context benchmarks, especially under reinforcement learning; notably, it outperforms existing long-context post-training corpora while using roughly a third of the samples. Finally, the generation pipeline extends naturally beyond titles, authors, and references to richer article elements (e.g., tables and figures) and to other domains whose entities have relational structure (e.g., patient records, financial reports). Because difficulty scales naturally with the underlying metadata, SciTrek can grow to probe increasingly sophisticated reasoning as context windows lengthen.

References

Chenxin An, Shansan Gong, Ming Zhong, Xingjian Zhao, Mukai Li, Jun Zhang, Lingpeng Kong, and Xipeng Qiu. L-eval: Instituting standardized evaluation for long context language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14388–14411, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.776. URL https://aclanthology.org/2024.acl-long.776/.

Akari Asai, Jacqueline He, Rulin Shao, Weijia Shi, Amanpreet Singh, Joseph Chee Chang, Kyle Lo, Luca Soldaini, Sergey Feldman, Mike D'Arcy, David Wadden, Matt Latzke, Minyang Tian, Pan Ji, Shengyan Liu, Hao Tong, Bohao Wu, Yanyu Xiong, Luke Zettlemoyer, Graham Neubig, Daniel S. Weld, Doug Downey, Wen-tau Yih, Pang Wei Koh, and Hannaneh Hajishirzi. Openscholar: Synthesizing scientific literature with retrieval-augmented lms. *CoRR*, abs/2411.14199, 2024. doi: 10.48550/ARXIV.2411.14199. URL https://doi.org/10.48550/arXiv.2411.14199.

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. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 3119–3137. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.172. URL https://doi.org/10.18653/v1/2024.acl-long.172.

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. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pp. 3639–3664. Association for Computational Linguistics, 2025. URL https://aclanthology.org/2025.acl-long.183/.

Guanzheng Chen, Michael Qizhe Shieh, and Lidong Bing. LongRLVR: Long-context reinforcement learning requires verifiable context rewards. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=omVhYvyTPJ.

DeepSeek-AI, Aixin Liu, Bei Feng, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, Chengqi Dengr, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Hao Yang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jin Chen, Jingyang Yuan, Junjie Qiu, Junxiao Song, Kai Dong, Kaige Gao, Kang Guan, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruizhe Pan, Runxin Xu, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Size Zheng, T. Wang, Tian Pei, Tian Yuan, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Liu, Xin Xie, Xingkai Yu, Xinnan Song, Xinyi Zhou, Xinyu Yang, Xuan Lu, Xuecheng Su, Y. Wu, Y. K. Li, Y. X. Wei, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Zheng, Yichao Zhang, Yiliang Xiong, Yilong Zhao, Ying He, Ying Tang, Yishi Piao, Yixin Dong, Yixuan Tan, Yiyuan Liu, Yongji Wang, Yongqiang Guo, Yuchen Zhu, Yuduan Wang, Yuheng Zou, Yukun Zha, Yunxian Ma, Yuting Yan, Yuxiang You, Yuxuan Liu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhewen Hao, Zhihong Shao, Zhiniu Wen, Zhipeng Xu, Zhongyu Zhang, Zhuoshu Li, Zihan Wang, Zihui Gu, Zilin Li, and Ziwei Xie. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model, 2024. URL https://arxiv.org/abs/2405.04434.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2368–2378, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1246. URL https://aclanthology.org/N19-1246/.

Gemini Team. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *Technical Report*, 2025. URL https://storage.googleapis.com/deepmind-media/gemini/gemini_v2_5_report.pdf.

Alexander Gurung and Mirella Lapata. Learning to reason for long-form story generation. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=dr3eg5ehR2.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021a. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In Joaquin Vanschoren and Sai-Kit Yeung (eds.), *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*, 2021b. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/be83ab3ecd0db773eb2dc1b0a17836a1-Abstract-round2.html.

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? *CoRR*, abs/2404.06654, 2024. doi: 10.48550/ARXIV.2404.06654. URL https://doi.org/10.48550/arXiv.2404.06654.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, et al. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*, 2024.

Greg Kamradt. Needle in a haystack - pressure testing llms. https://github.com/gkamradt/LLMTest_NeedleInAHaystack, 2023. URL https://github.com/gkamradt/LLMTest_NeedleInAHaystack.

Yuri Kuratov, Aydar Bulatov, Petr Anokhin, Ivan Rodkin, Dmitry Sorokin, Artyom Y. Sorokin, and Mikhail Burtsev. Babilong: Testing the limits of llms with long context reasoning-in-a-haystack. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/c0d62e70dbc659cc9bd44cbcf1cb652f-Abstract-Datasets_and_Benchmarks_Track.html.

Jinyang Li, Binyuan Hui, Ge Qu, Jiaxi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo, Xuanhe Zhou, Ma Chenhao, Guoliang Li, Kevin Chang, Fei Huang, Reynold Cheng, and Yongbin Li. Can llm already serve as a database interface? a big bench for large-scale database grounded text-to-sqls. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 42330–42357. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/83fc8fab1710363050bbd1d4b8cc0021-Paper-Datasets_and_Benchmarks.pdf.

Mo Li, Songyang Zhang, Yunxin Liu, and Kai Chen. Needlebench: Can llms do retrieval and reasoning in 1 million context window? *CoRR*, abs/2407.11963, 2024. doi: 10.48550/ARXIV.2407.11963. URL https://doi.org/10.48550/arXiv.2407.11963.

Jiaheng Liu, Dawei Zhu, Zhiqi Bai, Yancheng He, Huanxuan Liao, Haoran Que, Zekun Wang, Chenchen Zhang, Ge Zhang, Jiebin Zhang, Yuanxing Zhang, Zhuo Chen, Hangyu Guo, Shilong Li, Ziqiang Liu, Yong Shan, Yifan Song, Jiayi Tian, Wenhao Wu, Zhejian Zhou, Ruijie Zhu, Junlan Feng, Yang Gao, Shizhu He, Zhoujun Li, Tianyu Liu, Fanyu Meng, Wenbo Su, Yingshui Tan, Zili Wang, Jian Yang, Wei Ye, Bo Zheng, Wangchunshu Zhou, Wenhao Huang, Sujian Li, and Zhaoxiang Zhang. A comprehensive survey on long context language modeling. *CoRR*, abs/2503.17407, 2025. doi: 10.48550/ARXIV.2503.17407. URL https://doi.org/10.48550/arXiv.2503.17407.

Llama Team. The llama 4 herd: The beginning of a new era of natively multimodal ai innovation. *Technical Report*, 2025. URL https://ai.meta.com/blog/llama-4-multimodal-intelligence/.

Seiji Maekawa, Hayate Iso, and Nikita Bhutani. Holistic reasoning with long-context lms: A benchmark for database operations on massive textual data. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=5LXcoDtNyq.

Youssef Mroueh. Reinforcement learning with verifiable rewards: Grpo's effective loss, dynamics, and success amplification, 2025. URL https://arxiv.org/abs/2503.06639.

OpenAI. Introducing gpt-4.1 in the api. *Technical Report*, 2025. URL https://openai.com/index/gpt-4-1/.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300, 2024. doi: 10.48550/ARXIV.2402.03300. URL https://doi.org/10.48550/arXiv.2402.03300.

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. *CoRR*, abs/2512.12967, 2025. doi: 10.48550/ARXIV.2512.12967. URL https://doi.org/10.48550/arXiv.2512.12967.

Chonghua Wang, Haodong Duan, Songyang Zhang, Dahua Lin, and Kai Chen. Ada-leval: Evaluating long-context llms with length-adaptable benchmarks. In Kevin Duh, Helena Gómez-Adorno, and Steven Bethard (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pp. 3712–3724. Association for Computational Linguistics, 2024a. doi: 10.18653/V1/2024.NAACL-LONG.205. URL https://doi.org/10.18653/v1/2024.naacl-long.205.

Lei Wang, Shan Dong, Yuhui Xu, Hanze Dong, Yalu Wang, Amrita Saha, Ee-Peng Lim, Caiming Xiong, and Doyen Sahoo. Mathhay: An automated benchmark for long-context mathematical reasoning in llms. *CoRR*, abs/2410.04698, 2024b. doi: 10.48550/ARXIV.2410.04698. URL https://doi.org/10.48550/arXiv.2410.04698.

Minzheng Wang, Longze Chen, Fu Cheng, Shengyi Liao, Xinghua Zhang, Bingli Wu, Haiyang Yu, Nan Xu, Lei Zhang, Run Luo, Yunshui Li, Min Yang, Fei Huang, and Yongbin Li. Leave no document behind: Benchmarking long-context llms with extended multi-doc QA. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*, pp. 5627–5646. Association for Computational Linguistics, 2024c. doi: 10.18653/V1/2024.EMNLP-MAIN.322. URL https://doi.org/10.18653/v1/2024.emnlp-main.322.

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. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=o29E01Q6bv.

Tomer Wolfson, Mor Geva, Ankit Gupta, Yoav Goldberg, Matt Gardner, Daniel Deutch, and Jonathan Berant. Break it down: A question understanding benchmark. *Trans. Assoc. Comput. Linguistics*, 8: 183–198, 2020. doi: 10.1162/TACL_A_00309. URL https://doi.org/10.1162/tacl_a_00309.

Tomer Wolfson, Harsh Trivedi, Mor Geva, Yoav Goldberg, Dan Roth, Tushar Khot, Ashish Sabharwal, and Reut Tsarfaty. Monaco: More natural and complex questions for reasoning across dozens of documents. *CoRR*, abs/2508.11133, 2025. doi: 10.48550/ARXIV.2508.11133. URL https://doi.org/10.48550/arXiv.2508.11133.

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. Longmemeval: Benchmarking chat assistants on long-term interactive memory. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025a. URL https://openreview.net/forum?id=pZiyCaVuti.

Xiaodong Wu, Minhao Wang, Yichen Liu, Xiaoming Shi, He Yan, Lu Xiangju, Junmin Zhu, and Wei Zhang. LIFBench: Evaluating the instruction following performance and stability of large language models in long-context scenarios. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 16445–16468, Vienna, Austria, July 2025b. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.803. URL https://aclanthology.org/2025.acl-long.803/.

An Yang, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoyan Huang, Jiandong Jiang, Jianhong Tu, Jianwei Zhang, Jingren Zhou, Junyang Lin, Kai Dang, Kexin Yang, Le Yu, Mei Li, Minmin Sun, Qin Zhu, Rui Men, Tao He, Weijia Xu, Wenbiao Yin, Wenyuan Yu, Xiafei Qiu, Xingzhang Ren, Xinlong Yang, Yong Li, Zhiying Xu, and Zipeng Zhang. Qwen2.5-1m technical report. *CoRR*, abs/2501.15383, 2025. doi: 10.48550/ARXIV.2501.15383. URL https://doi.org/10.48550/arXiv.2501.15383.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1259. URL https://aclanthology.org/D18-1259/.

Howard Yen, Tianyu Gao, Minmin Hou, Ke Ding, Daniel Fleischer, Peter Izsak, Moshe Wasserblat, and Danqi Chen. HELMET: How to evaluate long-context models effectively and thoroughly. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=293V3bJbmE.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3911–3921, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1425. URL https://aclanthology.org/D18-1425/.

A Numerical Reasoning in Datasets with Expert-Written Questions

Our numerical reasoning questions are constructed around atomic SQL operations, specifically sorting, filtering, and aggregating, and operators covering comparison, arithmetic and logical reasoning, shown in Table 3. These capabilities are fundamental building blocks of complex, realistic tasks. QDMR (Wolfson et al., 2020) provides a widely used formalism, in which complex questions can be decomposed into atomic sub-questions that require filtering, aggregating, comparing, sorting, logical and arithmetic operations. More recently, MoNaCo (Wolfson et al., 2025) demonstrates through manual decomposition of naturally occurring human-written questions that real-world complex questions in multi-document understanding frequently demand aggregation and arithmetic skills.

Our manual analysis of human-written or human-audited benchmark questions in OpenScholar (Asai et al., 2024), LongBench v2 (Bai et al., 2025), and Loong (Wang et al., 2024c) reveals that these questions implicitly require the same numerical reasoning capabilities targeted by SciTrek. Example questions are shown in Table 12. Notably, such capabilities arise even in expert-written financial questions from LongBench v2, underscoring their relevance well beyond scientific text.

Index	Question	Operations	Why
Q1-OS	What are the latest works on finetuning an auto-regressive LM for dense passage retrieval, and how does their performance compare with bi-directional encoders?	sorting, comparing	Order works by recency; compare reported performance across two model classes.
Q2-OS	Which downstream task can be solved by AlphaFold3 but cannot be performed by ESM-3?	filtering, logical	Filter tasks by capability under a conjunction with negation.
Q3-OS	Compared to 2023, how has the percentage of finished-goods apparel factories from countries other than Vietnam, China, and Cambodia changed in 2024?	filtering, aggregating, arithmetic, comparing	Filter by excluded countries; aggregate counts; compute percentages; compare across years.
Q4-LB	In the financial reports of Apple and Samsung for 2022 and 2023, which company has a higher percentage of revenue from phones, and in what range do the differences between the two companies over the two years fall?	filtering, aggregating, arithmetic, comparing	Filter by company, year, and category; aggregate revenue; compute ratios and differences; compare.
Q5-LG	[We ask you to] review the provided papers and construct a citation chain [...] that accurately reflects the sequential citation order among these documents.	sorting, filtering	Filter to citing–cited pairs; order them into a sequential chain.

Table 12: Expert-written questions from OpenScholar (OS), LongBench v2 (LB), and Loong (LG), reproduced verbatim (our elisions marked [...]), with the SciTrek operations (Table 3) each implicitly requires and a brief justification. These illustrate that the operations SciTrek targets recur in naturally occurring benchmark questions.

B SciTrek Database Construction

We construct a database with three tables for each article collection, using metadata from Semantic Scholar along with basic preprocessing. A description of these tables is provided in Table 13. Our test dataset covers all SQL commands and operators that are listed in Table 3. The detailed distribution is shown in Table 14.

Database Table	Column Name	Data Type	Description
articles	article_id	String	The unique identifier of the article
	article_title	String	The title of the article
	title_word_count	Integer	The number of words in the article's title (using spaces to determine word boundaries)
	author_count	Integer	The number of authors in the article
	reference_count	Integer	The number of references that are cited in the article
article-author	relation_id	String	The unique identifier of the article-author relations
	article_id	String	The identifier of the associated article
	author_name	String	The name of the author/s
	author_position	Integer	The position of the author in the author list (starting from 0 for the first author)
citing-cited	relation_id	String	The unique identifier of the citation relations between two articles
	article_id_citing	String	The identifier of the article that cites another article
	article_id_cited	String	The identifier of the article that is cited by another article

Table 13: Description of database tables used to curate SQL queries and answers.

Command/Operator	Count	%	Command/Operator	Count	%
SELECT	2,121	100.00	GROUP BY	175	8.25
WHERE	1,821	85.86	$\leq$	175	8.25
$=$	1,032	48.66	%	164	7.73
IN	682	32.15	$\geq$	154	7.26
OR	616	29.04	NOT	154	7.26
ORDER BY	591	27.86	$\neq$	153	7.21
$<$	476	22.44	AVG	135	6.36
$>$	450	21.22	MIN	133	6.27
COUNT	370	17.44	BETWEEN	106	5.00
ASC	301	14.19	SUM	102	4.81
DESC	287	13.53	/	100	4.71
DISTINCT	280	13.20	LIKE	50	2.36
*	215	10.14	+	48	2.26
MAX	213	10.04	$-$	23	1.08
AND	195	9.19			

Table 14: Distribution of SQL commands and operators in our test data. Percentages reflect the prevalence of each command or operator among SQL queries and need not add to 100%.

We use LLMs to extract titles, authors, and references from article texts; the corresponding prompts are shown in Appendix F, Figure 3.

We generate natural language questions by prompting Qwen2.5-Coder-32B to convert SQL queries into natural language; the corresponding prompt is shown in Figure 4. The prompt for the reverse conversion, from natural language questions back to SQL queries, is shown in Figure 5.

C Experimental Settings and Evaluation

Configuration details for all models are presented in Table 15, with default inference settings from their Huggingface repositories used throughout.

Our evaluation covers both thinking and non-thinking models. For non-thinking models, we use two prompts: one to assess LLM capabilities against full-text articles (Figure 6) and one for using the corresponding database tables as context (Figure 8).

The thinking models are Qwen3-4B-Thinking-2507, Qwen3-30B-A3B-Thinking-2507, DeepSeek-R1-Distill-Llama-70B, Gemini 2.5 Pro, and o4-mini, which differ in their thinking configurations: o4-mini uses the default *medium* thinking effort with non-thinking prompts (Figure 6 and Figure 8), Gemini 2.5 Pro is allocated a budget of 512 tokens with non-thinking prompts (Figure 6 and Figure 8), and the three open-

Model	#Params	Think	Context Size	Release Date
Open-weight				
Qwen2.5-7B-Instruct-1M	7B	✗	1,010,000	Jan 2025
Qwen2.5-14B-Instruct-1M	14B	✗	1,010,000	Jan 2025
Qwen3-4B-Instruct-2507	4B	✗	262,144	Jul 2025
Qwen3-4B-Thinking-2507	4B	✓	262,144	Jul 2025
Qwen3-30B-A3B-Instruct-2507	30B (3B)	✗	262,144	Jul 2025
Qwen3-30B-A3B-Thinking-2507	30B (3B)	✓	262,144	Jul 2025
Gemma-3-27B-IT	27B	✗	131,072	Mar 2025
Llama-4-Scout-17Bx16E-Instruct	109B (17B)	✗	10,485,760	Apr 2025
Llama-3.3-70B-Instruct	70B	✗	131,072	Dec 2024
DeepSeek-R1-Distill-Llama-70B	70B	✓	131,072	Jan 2025
Proprietary				
GPT-4.1	—	✗	1,047,576	Apr 2025
Gemini 2.5 Pro	—	✓	1,048,576	Jun 2025
o4-mini	—	✓	200,000	Apr 2025

Table 15: LLMs evaluated in our experiments. **#Params**: total parameters, with active parameters per token in brackets for mixture-of-experts models; parameter counts are not disclosed for proprietary models. **Think** (✓, shaded rows): models prompted with chain-of-thought (Section 4). **Context Size**: maximum number of input tokens supported.

source models were prompted with the thinking prompt in Figure 7 which allows models to output visible reasoning steps before the answer.

Finally, the prompt used by GRPO to generate reasoning traces is given in Figure 7.

			Full-text Articles				Database Tables			
Model	**Ctx**	**Think**	**F-64**	**F-128**	**F-512**	**F-1024**	**D-64**	**D-128**	**D-512**	**D-1024**
Qwen-2.5-Coder + SQL execution	—	—	51.5	49.9	50.8	52.7	94.0	91.8	89.4	91.6
Qwen2.5-7B-Instruct-1M	1M	✗	8.7	7.0	1.1	0.0	39.2	39.3	28.7	21.2
Qwen2.5-14B-Instruct-1M	1M	✗	20.6	17.5	7.4	1.0	58.2	53.9	44.8	34.8
Qwen3-4B-Instruct-2507	256K	✗	7.2	7.3	—	—	47.2	46.0	38.9	31.1
Qwen3-4B-Thinking-2507	256K	✓	50.2	42.3	—	—	92.7	85.8	79.4	63.6
Qwen3-30B-A3B-Instruct-2507	256K	✗	15.0	16.4	—	—	55.6	53.8	44.7	42.0
Qwen3-30B-A3B-Thinking-2507	256K	✓	62.5	55.1	—	—	94.4	88.3	80.8	74.5
Gemma-3-27B-IT	128K	✗	22.6	14.7	—	—	52.8	52.5	47.8	40.7
Llama-4-Scout-17Bx16E-Instruct	10M	✗	20.3	17.4	15.6	14.8	49.8	44.9	40.5	37.4
Llama-3.3-70B-Instruct	128K	✗	25.3	15.1	—	—	65.8	63.5	53.7	49.5
DeepSeek-R1-Distill-Llama-70B	128K	✓	29.0	9.0	—	—	85.5	77.6	68.0	62.3
GPT-4.1	1M	✗	36.0	29.7	22.3	19.6	82.8	73.3	61.8	56.3
Gemini 2.5 Pro	1M	✓	58.1	48.8	⋆	⋆	95.3	88.3	79.0	69.8
o4-mini	195K	✓	74.0	63.8	—	—	97.2	89.2	87.0	86.5

Table 16: F1 (%) on SciTrek for the SQL execution-based method (top), open-weight (middle), and proprietary (bottom) models, under two context settings: full-text scientific articles and the corresponding database tables. **Ctx** is the maximum input length each model supports; **Think** (✓, shaded rows) marks models prompted with chain-of-thought, while non-thinking models (✗) answer directly. F-* denotes test instances using full-text article collections of up to 64K, 128K, 512K, and 1M tokens. D-* denotes test data instances using the database tables corresponding to F-64, 128, 512, and 1024. For any given article collection, the database representation is substantially shorter (mean ∼2K tokens), and can therefore be passed to all models regardless of their context limit. — : the model cannot process the given context size; ⋆: not evaluated due to prohibitive computational cost. Exact match results are in Table 7.

D Evaluation Results in Terms of F1

Table 16 reports results under F1. The ranking of models is unchanged from exact match (Table 7), confirming that our findings are not an artefact of a strict matching criterion, but absolute scores are substantially higher (e.g., o4-mini improves from 46.5 to 63.8 at 128K tokens with full-text articles). The gap between the two metrics is largest for questions with list-valued answers, as a sorted list with a single incorrect element receives a score of zero in terms of exact match, regardless of how close the answer is. This indicates that models frequently retrieve most of a required set or list while failing to produce it exactly. Two patterns from the main results also persist under F1: thinking models outperform their non-thinking counterparts by a wide margin, and performance degrades with context length for every model. The SQL execution-based method is the sole exception: because it processes articles in 16K-token chunks, its performance is essentially flat across collection sizes.

E Failure Patterns

This appendix provides the per-skill breakdowns behind the four failure patterns summarised in Section 4. All figures are computed over the 60 manually inspected instances at 128K input length (10 per skill in Table 4) on which every zero-shot and post-trained model fails, except for Table 18, which reports exact match over all 132 negation instances in the test set.

Fallback to NULL and format non-compliance. Table 17 reports both patterns by skill. NULL fallback is concentrated in the smaller open-weight models (Qwen2.5-7B: 90% on Aggregating, Qwen3-4B-Instruct: 90% on Filtering+Sorting) and is markedly rarer for thinking models and proprietary systems; post-training removes it almost entirely (0% in every cell for GRPO). Format violations follow a different distribution: they are worst for GPT-4.1 (30–50% across Aggregating, Sorting, and Filtering) and Gemma-3-27B, and, unlike NULL fallback, do not correlate with model size.

Negation. Our test set contains 132 negation instances, distributed across Filtering (11), Filtering+Aggregating (15), Filtering+Sorting (12), and Relational Filtering (94). Table 18 shows exact match on this subset: seven of thirteen models score below 5% overall, and no model exceeds 19%. The Relational Filtering column, which carries most instances, is where scores are lowest for every model ($\leq$6.4%), indicating that negation compounds with relational reasoning rather than being an independent difficulty.

Hallucination of specific facts. Inspecting the reasoning traces of Qwen3-4B-Thinking-2507, Qwen3-30B-A3B-Thinking-2507, and DeepSeek-R1-Distill-Llama-70B, we find that the high-level plans are typically correct (e.g., identify the relevant articles, count the target items, compare the counts) while the intermediate quantities are not. Counting reference items is the most common point of failure: models state a specific reference count for an article and proceed to reason correctly from that erroneous value.

F Prompt Templates

This appendix collects the prompt templates used throughout SciTrek. Placeholders instantiated at run time are marked in red. Figure 3 contains the four templates used to extract article metadata from full text: title extraction, author extraction, reference counting, and detection of citation relationships between article pairs. Each is applied to 16K-token chunks of an article, as the extraction model (Qwen2.5-Coder-32B-Instruct) does not reliably handle longer inputs; for references we count entries and detect citation links separately rather than reproducing the full reference list, which is more robust than extracting bibliographies verbatim.

Figures 4 and 5 contain the templates for the two directions of query-question conversion. Both present the same database schema, so that a question generated from an SQL query in Figure 4 can be converted back to SQL under identical assumptions in Figure 5; this round-trip allows us to verify that generated questions preserve the meaning of the original query (Section 3). All prompts require JSON output, which we enforce with constrained decoding.

	Fallback to NULL						Format Non-compliance					
Model	**A**	**S**	**F**	**F+A**	**F+S**	**R**	**A**	**S**	**F**	**F+A**	**F+S**	**R**
Qwen2.5-7B-Instruct-1M	90	10	80	10	40	70	0	0	0	0	0	10
Qwen2.5-14B-Instruct-1M	0	0	30	30	30	80	10	30	0	20	30	10
Qwen3-4B-Instruct-2507	50	60	80	10	90	90	0	0	0	0	0	0
Qwen3-4B-Thinking-2507	0	0	20	20	10	40	0	0	0	0	0	0
Qwen3-30B-A3B-Inst.-2507	0	0	0	20	0	70	30	0	50	40	10	10
Qwen3-30B-A3B-Think.-2507	0	0	0	30	10	30	0	10	0	0	0	0
Gemma-3-27B-IT	10	0	0	40	10	40	20	30	40	10	0	30
Llama-4-Scout-Instruct	10	10	10	60	30	80	10	30	10	20	0	0
Llama-3.3-70B-Instruct	20	30	40	70	30	70	0	10	10	10	10	10
DeepSeek-R1-Dist.-Llama-70B	0	0	10	10	0	0	0	0	0	0	0	0
GPT-4.1	0	0	0	0	0	10	50	40	40	30	10	10
Gemini 2.5 Pro	0	0	0	20	0	30	10	0	0	10	0	0
o4-mini	0	0	0	30	10	20	10	10	0	0	0	0
Qwen2.5-7B (SFT)	0	0	20	70	20	0	0	0	0	0	0	0
Qwen2.5-7B (GRPO)	0	0	0	0	0	0	10	0	0	0	10	0

Table 17: Proportion of inspected samples (%) exhibiting the two output-level failure patterns, by information-processing skill. **A**: Aggregating; **S**: Sorting; **F**: Filtering; **R**: Relational Filtering. Shaded rows are thinking models; the final block shows post-trained variants of Qwen2.5-7B-Instruct-1M.

Model	**F**	**F+A**	**F+S**	**R**	**All**
Qwen2.5-7B-Instruct-1M	0	0	0	0	0
Qwen2.5-14B-Instruct-1M	0	20.0	0	0.2	3.8
Qwen3-4B-Instruct-2507	0	0	0	0	0
Qwen3-4B-Thinking-2507	0	53.3	8.3	1.1	9.1
Qwen3-30B-A3B-Inst.-2507	0	0	0	0	0
Qwen3-30B-A3B-Thin.-2507	0	66.7	16.7	1.1	12.9
Gemma-3-27B-IT	0	6.7	0	1.1	1.5
Llama-4-Scout-Instruct	0	13.3	0	2.1	3.0
Llama-3.3-70B-Instruct	0	6.7	0	0	0.8
DeepSeek-R1-Dist.-Llama-70B	0	13.3	0	0	3.0
Gemini 2.5 Pro	0	20.0	8.3	6.4	9.8
GPT-4.1	0	13.3	0	3.2	4.5
o4-mini	0	60.0	50.0	5.3	18.9
Qwen2.5-7B-Inst.-1M (SFT)	0	6.7	0	10.6	9.1
Qwen2.5-7B-Inst.-1M (GRPO)	0	33.3	8.3	4.3	8.3

Table 18: Model performance on questions involving negation (e.g., “not" or “never"), reported in terms of exact match (%). The first block shows zero-shot models; the second block shows supervised versions of Qwen2.5-7B-Instruct-1M. **A** abbreviates Aggregation, **S** stands for Sorting, **F** for Filtering, and **R** for Relational Filtering.

Figures 6 and 8 contain the templates used to evaluate models on SciTrek. The two differ only in how the context is presented: the first supplies the full-text articles, the second the corresponding database tables, while the answer-format instructions are identical, so that the two settings differ in input representation alone. Figure 7 gives the variant used for thinking models and for GRPO training, which additionally asks the model to reason step by step and place its final answer in `\boxed{}`; it is otherwise identical to the non-thinking prompt.

Instruction for Title Extraction

You are given a chunk of a scientific article.
{context}
If the article title is included in the given text, please locate and extract the title.
You must output the title in the following JSON format, and do not include any extra text.
{"title": "the extracted title"}
If there is no article title in the given text, just output {"title": ""}

Instruction for Author Extraction

You are given a chunk of a scientific article.
{context}
If the author list of the article is included in the given text, please locate and extract the author list.
You must output the author list in the following JSON format, and do not include any extra text.
{"authors": "the author list separated with commas"}
If there is no author list in the given text, just output {"authors": ""}

Instruction for Counting References

You are given a chunk of a scientific article.
{context}
If a complete or partial reference section is included in the given text, please count the number of references.
You must output the reference count in the following JSON format, and do not include any extra text.
{"reference_count": "the number of references"}
If there is no reference section in the given text, just output {"reference_count": "0"}

Instruction for Detecting Citation Relationships

You are given the title of one scientific article (Article A) and a chunk of another scientific article (Article B).
The title of Article A: {title}
The chunk of Article B: {context}
Please check whether Article A is cited by Article B; that is, check whether the title of Article A appears in the given chunk of Article B.
You must output your answer in the following JSON format, and do not include any extra text.
{"a_cited_by_b": "true or false"}

Figure 3: Prompt templates for extracting titles, authors, and references from article text. {context} and {title} are placeholders for the input article chunk and the article title.

Instruction to Convert SQL Queries to Natural Language Questions

You are given a database with three tables: *articles*, *article-author*, and *citing-cited*.

The *articles* table contains the following columns:

- article_id (String): the unique identifier of the article;
- article_title (String): the title of the article;
- title_word_count (Integer): the number of words in the article's title (using spaces to determine word boundaries);
- author_count (Integer): the number of authors for the article;
- reference_count (Integer): the number of references cited in the article.

The *article-author* table contains the following columns:

- relation_id (String): the unique identifier of the article-author relationship;
- article_id (String): the identifier of the associated article;
- author_name (String): the name of the author;
- author_position (Integer): the position of the author in the author list (starting from 0 for the first author).

The *citing-cited* table contains the following columns:

- relation_id (String): the unique identifier of the citation relationship between two articles;
- article_id_citing (String): the identifier of the article which cites the other article;
- article_id_cited (String): the identifier of the article which is cited by the other article.

Assumptions:

- The *articles* table contains multiple entries;
- The *article_author* table maps authors to articles, where one author can contribute to multiple articles, and one article can have multiple authors;
- The *citing_cited* table represents citation relationships among articles in the articles table, where one article can be cited by multiple others.

Your task involves two steps:

1. Understand the given SQL query in the context of the database schema described above;
2. Convert the SQL query into a clear and natural-sounding question in everyday language, as if you were reading textual articles rather than querying a database.

The given SQL query: {sql_query}

Do not refer to relation_id or article_id in the natural-language question.
You must output the SQL query and the corresponding question in the following JSON format, and do not include any extra text:

{"sql": "the given SQL query", "question": "the generated question"}

Figure 4: Prompt template for converting SQL queries to natural language questions.

Instruction to Convert Natural Language Questions to SQL Queries

You are given a database with three tables: *articles*, *article-author*, and *citing-cited*.

The *articles* table contains the following columns:

- article_id (String): the unique identifier of the article;
- article_title (String): the title of the article;
- title_word_count (Integer): the number of words in the article's title (using spaces to determine word boundaries);
- author_count (Integer): the number of authors for the article;
- reference_count (Integer): the number of references cited in the article.

The *article-author* table contains the following columns:

- relation_id (String): the unique identifier of the article-author relationship;
- article_id (String): the identifier of the associated article;
- author_name (String): the name of the author;
- author_position (Integer): the position of the author in the author list (starting from 0 for the first author).

The *citing-cited* table contains the following columns:

- relation_id (String): the unique identifier of the citation relationship between two articles;
- article_id_citing (String): the identifier of the article which cites the other article;
- article_id_cited (String): the identifier of the article which is cited by the other article.

Assumptions:

- The *articles* table contains multiple entries;
- The *article_author* table maps authors to articles, where one author can contribute to multiple articles, and one article can have multiple authors;
- The *citing_cited* table represents citation relationships among articles in the articles table, where one article can be cited by multiple others.

Available core SQL commands:

- Aggregating: `MIN()`, `MAX()`, `COUNT()`, `SUM()`, `AVG()`, `DISTINCT`
- Filtering: `WHERE`
- Organizing: `ORDER BY`, `ASC`, `DESC`, `GROUP BY`

Available core SQL operators:

- Comparison: $=$, $>$, $<$, $>=$, $<=$, $<>$, `LIKE`
- Arithmetic: $+$, $-$, $*$, $/$, $\%$
- Logical: `AND`, `NOT`, `OR`, `BETWEEN`, `IN`

Your task is to:

1. Understand the database schema described above and the given natural language question below;
2. Convert the natural language question into a SQL query in the context of the database schema with the listed SQL commands and operators.

The given natural language question: {question}

Do not output relation_id or article_id in generated SQL query.
Use the SQL commands and operators listed above.
Make the generated SQL query aligned well with the natural language question.
You must output the natural language question and the generated SQL query in the following JSON format, and do not include any extra text:
{"question": "the given question", "sql": "the generated SQL query"}

Figure 5: Prompt template for converting natural language questions to SQL queries.

Testing Instruction Using Full-text Articles

Articles:

{scientific articles}

You are provided with multiple scientific articles above. Based on the information in these articles, answer the question provided below.

If the answer consists of multiple components (e.g., author names, article titles, reference counts), separate them with commas.
For example, if the answer includes two author names, your response should be in the format of 'the-first-author-name, the-second-author-name'.
When counting the number of words in article titles, use spaces to determine word boundaries. Words are spaced apart individually.

Respond with only the final answer, with no additional explanation or formatting. If you cannot get the answer from the articles, just return 'NULL'.

Question:

{question}

Figure 6: Prompt template using full-text articles as context.

Instruction Template to Generate Reasoning Traces

Articles:

{scientific articles}

You are provided with multiple scientific articles above. Based on the information in these articles, answer the question provided below.

If the answer consists of multiple components (e.g., author names, article titles, reference counts), separate them with commas.
For example, if the answer includes two author names, your response should be in the format of 'the-first-author-name, the-second-author-name'.
When counting the number of words in article titles, use spaces to determine word boundaries. Words are spaced apart individually.

Think step by step, and place your final answer within \boxed{}. If you cannot get the answer from the articles, just return 'NULL'.

Question: {question}

Figure 7: Prompt template in the thinking mode given full-text articles or database tables as context.

Testing Instruction Using Database Tables

You are given three tables, named *articles*, *article-author*, and *citing-cited*.

The *articles* table contains the following columns:

- article_id (String): the unique identifier of the article;
- article_title (String): the title of the article;
- title_word_count (Integer): the number of words in the article's title (using spaces to determine word boundaries);
- author_count (Integer): the number of authors for the article;
- reference_count (Integer): the number of references cited in the article.

{the table of articles}

The *article-author* table contains the following columns:

- relation_id (String): the unique identifier of the article-author relationship;
- article_id (String): the identifier of the associated article;
- author_name (String): the name of the author;
- author_position (Integer): the position of the author in the author list (starting from 0 for the first author).

{the table of article-author}

The *citing-cited* table contains the following columns:

- relation_id (String): the unique identifier of the citation relationship between two articles;
- article_id_citing (String): the identifier of the article which cites the other article;
- article_id_cited (String): the identifier of the article which is cited by the other article.

{the table of citing-cited}

Based on the information in the tables above, answer the question provided below.

If the answer consists of multiple components (e.g., author names, article titles, reference counts), separate them with commas. For example, if the answer includes two author names, your response should be in the format of 'the-first-author-name, the-second-author-name'.

Respond with only the final answer, with no additional explanation or formatting. If you cannot get the answer from the tables, just return 'NULL'.

Question: {question}

Figure 8: Prompt template using database tables as context.