ARXIV2TABLE: Toward Realistic Benchmarking and Evaluation for LLM-Based Literature-Review Table Generation

Weiqi Wang[♣♠], **Jiefu Ou**[♣], **Yangqiu Song**[♠], **Benjamin Van Durme**[♣], **Daniel Khashabi**[♣]
[♣]Center for Speech and Language Processing, Johns Hopkins University, Baltimore, MD, USA
[♠]Department of Computer Science and Engineering, HKUST, Hong Kong SAR, China
{wwangbw, yqsong}@cse.ust.hk, {jou6, vandurme, danielk}@jhu.edu

Abstract

Literature review tables are essential for summarizing and comparing collections of scientific papers. In this paper, we study automatic generation of such tables from a pool of papers to satisfy a user's information need. Building on recent work (Newman et al., 2024), we move beyond oracle settings by (i) simulating *well-specified yet schema-agnostic* user demands that avoid leaking gold column names or values, (ii) explicitly modeling retrieval noise via semantically related but out-of-scope *distractor* papers verified by human annotators, and (iii) introducing a lightweight, annotation-free, utilization-oriented evaluation that decomposes utility (schema coverage, unary cell fidelity, pairwise relational consistency) and measures paper selection via a two-way QA procedure (gold→system and system→gold) with recall, precision, and F1. To support reproducible evaluation, we introduce ARXIV2TABLE, a benchmark of 1,957 tables referencing 7,158 papers, with human-verified distractors and rewritten, schema-agnostic user demands. We also develop an *iterative, batch-based* generation method that co-refines paper filtering and schema over multiple rounds. We validate the evaluation protocol with human audits and cross-evaluator checks. Extensive experiments show that our method consistently improves over strong baselines, while absolute scores remain modest, underscoring the task's difficulty. Our data and code is available at https://github.com/JHU-CLSP/arXiv2Table.

1 Introduction

Literature review tables play a crucial role in scientific research by organizing and summarizing large amounts of information from selected papers into a concise and comparable format (Russell et al., 1993). At the core of these tables are the *schema* and *values* that define their structure, where *schema* refers to the categories or aspects used to summarize different papers and *values* correspond to the specific information extracted from each paper. A well-defined *schema* allows each work to be represented as a row of *values*, enabling structured and transparent comparisons across different studies.

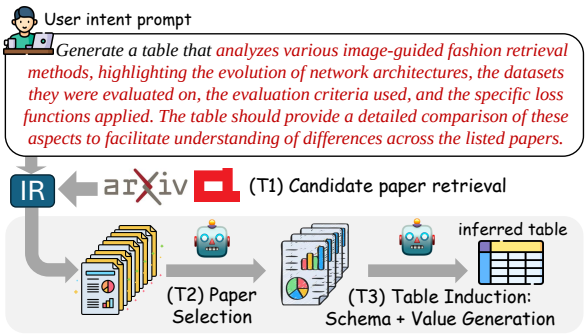

Figure 1: Overview of our proposed task: Given a user's demand, a simulated information retrieval (IR) engine first retrieves semantically relevant papers. Then, a language model further filters them and induces the table's corresponding schema and values to satisfy the user's demand. The grayed region indicates the scope covered by our method and benchmark (ARXIV2TABLE).

With recent advancements in large language models (LLMs; OpenAI, 2025c; DeepSeek-AI et al., 2025), several studies (Newman et al., 2024; Dagdelen et al., 2024; Sun et al., 2024) have explored generating literature review tables by prompting LLMs with a set of pre-selected papers and the table's caption. While these efforts represent meaningful progress, we argue that the existing task definition and evaluation protocols are somewhat unrealistic, thus hindering the practical applicability of generation methods.

First, existing pipelines assume that all provided papers are relevant and should be included in the table. However, in real-world scenarios, distractor papers—those that are irrelevant or contain limited useful information—are common (OpenAI, 2025a). Models should be able to identify and filter out such papers before table construction. Additionally, current pipelines use the ground-truth table's descrip-

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 7602–7624

tive caption as the objective for generation. These captions often lack sufficient context, making it difficult for LLMs to infer an appropriate schema, or they may inadvertently reveal the schema and values, leading to biased evaluations.

In this paper, we introduce our task, as illustrated in Figure 1, which improves upon previous task definitions through two key adaptations. First, our pilot study shows that LLMs struggle to retrieve relevant papers from large corpora. To benchmark this, we introduce distractor papers by selecting them based on semantic similarity to papers in the ground-truth table. LLMs must first determine which papers should be included before generating the table. Second, we simulate *well-specified yet schema-agnostic* user demands that describe the goal of curating the table without revealing column labels or cell values. This formulation is more realistic than terse captions while avoiding schema/value leakage that would bias evaluation. We build upon the ARXIVDIGESTABLES (Newman et al., 2024) dataset and construct a sibling benchmark through human annotation to verify the selected distractors, comprising 1,957 tables and 7,158 papers. Meanwhile, current evaluation methods rely on static semantic embeddings to estimate schema overlap between generated and ground-truth tables and require human annotations to assess the quality of unseen schemas and values. However, semantic embeddings struggle to capture nuanced, context-specific variations due to their reliance on pre-trained representations, while human annotation is costly and time-consuming. Moreover, the most effective table generation approaches define schemas primarily based on paper abstracts. This method risks missing important aspects present in the full text, leading to loosely defined schemas with inconsistent granularity.

To address these issues, we propose an annotation-free evaluation framework that instructs an LLM to synthesize QA pairs based on the ground-truth table and assess the generated table by answering these questions. These QA pairs evaluate table content overlap across three dimensions: schema-level, single-cell, and pairwise-cell comparisons. Additionally, we introduce a novel table generation method that batches input papers, iteratively refining paper selection and schema definition by revisiting each paper multiple times. Extensive experiments using five LLMs demonstrate that they struggle with both selecting relevant papers and generating high-quality tables, while our method significantly improves performance on both fronts. Expert validation further confirms the reliability of our QA-synthetic evaluations.

In summary, our contributions are threefold: (1) We introduce an improved task definition for literature review tabular generation, benchmarking it in a more realistic scenario by incorporating distractor papers and replacing table captions with abstract user demands; (2) We propose an annotation-free evaluation framework that leverages LLM-generated QA pairs to assess schema-level, single-cell, and pairwise-cell content overlap, addressing the limitations of static semantic embeddings and human evaluation; and (3) We develop a novel iterative batch-based table generation method that processes input papers in batches, refining schema definition and paper selection iteratively.

To the best of our knowledge, we are the first to introduce a task that simulates real-world use cases of scientific tabular generation by incorporating user demands and distractor papers, providing a more robust assessment of LLMs in this domain.

2 Related Works

Scientific literature tabular generation Prior works primarily attempt to generate scientific tables through two stages: schema induction and value extraction. For schema induction, early methods like entity-based table generation (Zhang and Balog, 2018) focused on structured input, while recent work has explored schema induction from user queries (Wang et al., 2024b) and comparative aspect extraction (Hashimoto et al., 2017). For value extraction, various approaches such as document-grounded question-answering (Kwiatkowski et al., 2019; Dasigi et al., 2021; Lee et al., 2023), aspect-based summarization (Ahuja et al., 2022), and document summarization (DeYoung et al., 2021; Lu et al., 2020) have been proposed to extract relevant information.

Beyond these methods, several datasets have been introduced to support scientific table-related tasks, such as TableBank (Li et al., 2020), SciGen (Moosavi et al., 2021), and SciTabQA (Lu et al., 2023). Ramu et al. (2024) propose an entailment-oriented evaluation complementary to our QA-based protocol. Recently, Newman et al. (2024) proposed streamlining schema and value generation with LLMs sequentially and curated a large-scale benchmark for evaluation. However, all these methods assume a clean and fully relevant

set of papers and rely on predefined captions or abstract-based schemas In contrast, we argue for an evaluation approach where candidate papers include tangentially relevant or distracting papers, aligning more closely with real-world literature review workflows (Padmakumar et al., 2025).

Table induction for general domains Other than the scientific domain, table induction is also widely studied as text-to-table generation. Prior works attempt this as a sequence-to-sequence task (Li et al., 2023; Wu et al., 2022) or as a question-answering problem (Sundar et al., 2024; Tang et al., 2023). Similar to these works, our framework is capable of better handling both structured and distractive input for real-world literature review and knowledge synthesis.

3 Task Definition

We first define a pipeline consisting of three subtasks that extend prior definitions and better capture the real-world usage of literature review tabular generation. For all the following tasks, we are given a user demand prompt p, which specifies the intended purpose of creating the table.

(T1) Candidate Paper Retrieval: We begin with a given *universe* of papers (e.g., the content of Google Scholar or arXiv) from which relevant papers need to be identified. Given a large collection, the goal is to use a search engine (IR) to retrieve a subset of *candidate* papers $C := \{d_i\}_{i=1}^{M}$ of size M, which may include distractor papers—i.e., papers that resemble the user demand prompt but do not fully satisfy the requirement.

(T2) Paper Selection: Given C, the second subtask is to select the *relevant* subset of size m ($m < M$): $R := \{d_i\}_{i=1}^{m} \subseteq C$, which best aligns with the user demand p. T2 differs from T1 in scale. Due to the large scale of T1, IR engines must optimize for recall, ensuring that as many relevant papers as possible are retrieved. However, T2 operates at a smaller scale, where precision is the priority, as it focuses on filtering out distractors and selecting only the most relevant papers.

(T3) Table Induction: Given the selected papers R, the objective is to generate a table with m rows and N columns, where $N \geq 2$ (i.e., no single-column tables). Each row $r_i \in \{r_1, r_2, \ldots, r_m\}$ corresponds to a unique input document $d_i \in R$, and each column $c_j \in \{c_1, c_2, \ldots, c_N\}$ represents a unique aspect of the documents. We refer to these N columns as the *schema* of the table and the $N \times m$ cells as the *values* of the table. The value of each cell is derived from its respective document according to the aspect defined by the corresponding column.

4 ARXIV2TABLE Construction

We then construct ARXIV2TABLE based on the ARXIVDIGESTABLES dataset which consists of literature tables (extracted from computer science papers) and their corresponding captions. We filter out tables that are structurally incomplete or lack full text for all referenced papers. As a result, we are left with 1,957 tables (with captions) which have rows referring to 7,158 papers. Our construction involves three pillars: user demand inference (§4.1), a simulated paper retrieval (§4.2) and evaluation through utilization (§4.3).

4.1 Constructing User Demand Prompts

We simulate well-specified yet schema-agnostic user demands p by rewriting original survey-table captions into user-demand-style prompts that better reflect how a researcher would request a comparison table. Each rewritten prompt is required to be (i) *self-contained*, meaning it is understandable without seeing the table, (ii) *goal-oriented*, meaning it clearly states the purpose of the table, and (iii) *schema/value non-leaking*, meaning it does not include gold column names, specific cell values, or direct paraphrases of them.

Table captions are not appropriate prompts While the input dataset contains one caption per table, collected from arXiv papers, these captions are meant to complement tables rather than fully describe them. As a result, they are generally concise. For example, a table caption might read: *"Performance comparison of different approaches,"* which is too vague to understand without seeing the table. Consequently, using table captions as prompts may not yield a well-defined task. A more contextually self-contained rewritten user demand might instead be: *"Draft a table that compares different knowledge editing methods, focusing on their performance on QA datasets."* Operationally, a caption is deemed "under-specified" when it cannot unambiguously determine a comparison goal without seeing either the gold schema or table body.

Our prompt construction To address this issue, we rewrite the captions of literature review tables into abstract yet descriptive user intentions

Prompt	Content	#Table ↓	#Tokens ↓
Caption	Schema	101 (5.2%)	1.2
	Value	46 (2.4%)	1.3
User Demand	Schema	14 (0.7%)	1.0
	Value	8 (0.4%)	1.0

Table 1: Overlap statistics between prompts (the original caption or our constructed user demand) and table content (schema or values). **#Table:** Number (and %) of tables with at least one token from table content overlapping with the prompt. **#Tokens:** Average count of overlapping tokens between table content and prompt.

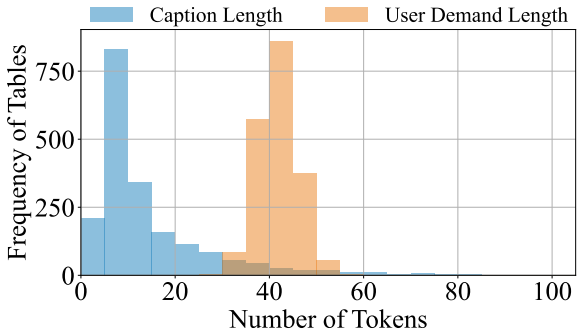

Figure 2: Distribution of the number of tokens between original captions and our modified user demands.

using GPT-4o. We guide the model with a prompt (see §B) that explains the rewriting task and specifies that the resulting user demand should be sufficiently contextualized to clearly state the table's purpose while avoiding the inclusion or direct description of column names or specific values. Here, the LLM is used solely for rewriting existing table captions into user demand prompts and for generating QA pairs grounded in ground-truth tables. These reformulations are strictly tied to observed data and do not require external factual knowledge, minimizing risks of contamination or model-specific bias.

To enforce our constraints, we apply an automatic leak check immediately after rewriting (Appendix B) and discard or rewrite any demand that reveals schema or value information. We also perform manual spot checks during construction to remove under-specified requests or prompts that implicitly reveal gold schema/value tokens. For simplicity, we collect only one user demand per table. More examples are provided in Appendix D.

Table captions vs. constructed user demand prompts To verify that our collected user demands align with our objective, we visualize: (1) the distribution of the number of tokens in the original and modified user demands, and (2) the ratio of captions and user demands of different lengths that have token overlap with the schema or values. From Figure 2, we observe that our modified user demands are generally longer than the original captions, providing a more detailed description of the table's goal. Furthermore, as shown in Table 1, user demands exhibit a significantly lower overlap ratio with the schema and table values, resulting in fewer overlapping tokens. Quantitative statistics on lexical leakage and overlap are reported in the main paper.

4.2 Paper Retrieval Simulation

The unreliability of paper retrieval Next, we approach the first subtask, candidate paper retrieval, by conducting a pilot study to assess whether LMs can reliably retrieve relevant papers from a large corpus. For each table, we employ a Sentence-BERT (Reimers and Gurevych, 2019) encoder as a retrieval engine, selecting papers from the entire corpus based on the highest similarity between the table's user demand and each paper's title and abstract. We vary the number of retrieved papers between 2 and 100 and plot the precision and recall of retrieval against the ground-truth papers in the original table (Figure 3).

We observe consistently low precision and recall across different retrieval sizes, highlighting the challenge of retrieving relevant papers from a noisy corpus. This demonstrates that the first subtask is non-trivial and may introduce noise into subtask T2. However, various information retrieval engines, such as Google Scholar and Semantic Scholar, can replace LMs in this subtask. Thus, we decide to simulate T1 by adding human-verified distractor papers into the candidate pool C, yielding a noisy input for T2 (paper selection on C to produce R). This allows us to focus on evaluating LLMs' capabilities in the T2 and T3 subtasks.

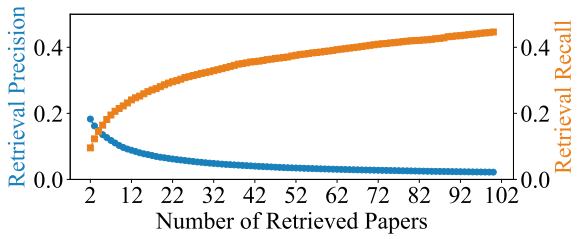

Figure 3: Precision and recall curves for different numbers of retrieved papers.

Similarity-based paper retrieval Moving forward, we associate distractor paper candidates with each table to simulate a potentially noisy document pool before constructing the table. Ideally, distractor candidates should be semantically related to the table but exhibit key differences that fail to meet the user demand. To select such candidates, we adopt a retrieve-then-annotate approach. First, we use a SentenceBERT encoder F to obtain embeddings for (1) the user demand $F(p)$ and (2) all papers in the corpus $\{F(d_i) \mid d_i \in U\}$. Each paper's embedding is computed by encoding the concatenation of its title and abstract. We then rank all papers $d_i \notin R$ based on the average of two cosine similarities: (1) the similarity between the candidate and the user demand, and (2) the average similarity between the candidate and each referenced paper:

$$s(d_i) = \cos(F(d_i), F(p)) + \frac{1}{m}\sum_{j=1}^{m} \cos(F(d_i), F(d_{u_j})).$$

Higher values of $s(d_i)$ indicate stronger semantic relevance, and we select the top 10 ranked papers for each table as its distractor candidates.

Candidates verification via human annotation After selecting these candidates, we conduct human annotations to verify whether they should indeed be excluded from the table. Given that annotating these tables requires expert knowledge in computer science, we recruit a trained team of annotators with research experience in the field as annotators. To ensure they are well-prepared for the task, the annotators undergo rigorous training, including pilot annotation exams. Their task is to make a binary decision on whether a given distractor paper—based on its title, abstract, user demand, the ground-truth table, and the titles and abstracts of all referenced papers—should be included in the table.

Each table contains annotations for 10 papers, with each distractor paper initially assigned to two randomly selected annotators. If both annotators agree on the label, it is finalized. Otherwise, two additional annotators review the paper until a consensus is reached. In the first round, the inter-annotator agreement (IAA) is 94% based on pairwise agreement, and the Fleiss' Kappa (Fleiss, 1971) score is 0.73, indicating a substantial level of agreement. Finally, for each table, we randomly select a number of distractor papers between $[m, 10]$ and merge them with R to form C.

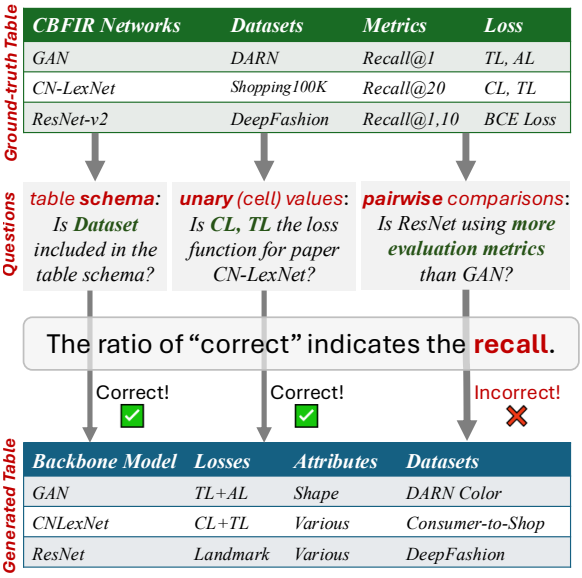

Figure 4: Overview of our proposed LLM-based QA-synthesis evaluation protocol, where LLMs synthesize QA pairs based on the ground-truth table and utilize the generated table to answer them. The ratio of successfully answered QA pairs indicate the ratio of information preserved.

4.3 Evaluation via LLM-based Utilization

After constructing the benchmark, we propose evaluating the quality of generated tables from a utilization perspective to address the challenge of aligning schemas and values despite potential differences in phrasing. This is achieved by synthesizing QA pairs based on the ground-truth table and using the generated table to answer them, or vice versa. The flexibility of this QA synthesis allows us to evaluate multiple dimensions of the table while ensuring a structured and scalable assessment. An overview with running examples is shown in Figure 4.

Dimensions of evaluating a table with QAs We introduce three key aspects for evaluating a table in terms of its usability: (1) **Schema**: whether a specific column is included in the generated schema, (2) **Unary Value**: whether a particular cell from the ground-truth table appears in the generated table, (3) **Pairwise Value**: whether relationships between two cells remain consistent in the generated table.

Recall evaluation We guide GPT-4o in generating binary QA pairs based on the ground-truth table. For the first two aspects, we generate QA pairs for all columns and cells, whereas for the third, we randomly sample 10 cell pairs per table and synthesize them into QA pairs. We then prompt GPT-4o to answer these questions based on the generated

table, providing yes/no responses. If the answer cannot be found, the model is instructed to respond with "no", and vice versa for "yes". The ratio of "yes" answers indicates how well the generated table preserves the schema, individual values, and pairwise relationships. This represents the **recall** of the ground-truth table, measuring how much original information is retained in generations.

Precision evaluation To additionally evaluate **precision**, we reverse the process: instead of generating QA pairs from the ground-truth table, we generate them from the generated table and ask another LLM (by default, we use the same backbone for consistency; cross-model swaps are reported in Appendix A.3) to answer them using the ground-truth table. The precision score reflects how much of the generated table's content is actually supported by the original data. By computing the ratio of "yes" answers, precision quantifies how much of the generated table is supported by the ground-truth table; novel content beyond gold is not credited under this automatic protocol. To assess evaluator dependence, we swap QA synthesizer/answerer model families and observe stable method rankings; results are in Appendix A.3.

5 Tabular Generation Methodologies

We explore a range of methods to evaluate on our proposed task, starting from several baselines inspired by prior work (§5.1) and then our proposed approach (§5.2).

5.1 Baseline Methods

We first introduce three methods for generating literature review tables to evaluate their performance on our task and use them as baselines for our proposed method. For easy reference, these methods are termed numerically.

First, **Baseline 1** generates the table in a one-step process. It takes all available papers R and the user demand p as input, and the model is asked to select all relevant papers and output a table with a well-defined schema and filled values in a single round of conversation. However, this method struggles with extremely long prompts that exceed the LLMs' context window when generating large tables.

To address this issue, **Baseline 2** processes papers individually. For each document, the model decides whether it should be included based on the user demand. If included, the model generates a table for that document. After processing all documents, the final table is created by merging the schemas of all individual tables using exact string matching and copying the corresponding values. While this approach reduces the input prompt length, it results in highly sparse tables due to inconsistent schema across papers and the potential omission of relevant information when individual papers lack sufficient context to define comprehensive table aspects.

To overcome both issues, Newman et al. (2024) introduce a two-stage process. In the first stage, the model selects papers relevant to the user demand based on their titles and abstracts, then generates a corresponding schema. In the second stage, the model loops through the selected papers and fills in the respective rows based on the full text of each document. A minor drawback of this method is that the schema is generated solely from titles and abstracts, which may overlook details present only in the full text. Note that this method is the **strongest recent baseline** for scientific tabular generation while other text-to-table methods (Deng et al., 2024b) are not directly applicable due to different assumptions.

To probe whether explicit reasoning mitigates multi-step synthesis errors, we also evaluate COT-augmented variants that add lightweight chain-of-thought style scaffolding to the strongest baseline and to our method.

5.2 Iterative Batch-based Tabular Generation

Then, we introduce our proposed method for generating literature review tables. Our approach consists of three steps: (A) key information extraction, (B) paper batching, and (C) paper selection and schema refinement, where the latter two steps can be iterated multiple times.

(A) Key Information Extraction Processing multiple papers simultaneously using their full text often results in excessively long prompts that exceed the LLMs' context window. To address this, we first shorten each paper by instructing the LLM to extract key information from the full text that is relevant to the user's requirements. Notably, we do not rely solely on the abstract, as important details often appear in the full text but are omitted from the abstract. For each paper, we provide the LLM with its title, abstract, and full text, along with the user's request, and ask it to generate a concise paragraph that preserves all potentially relevant details. These summary paragraphs serve as condensed represen-

Backbone Model	Method	Paper (T2)			Schema			Unary Value			Pairwise Value			Avg
		R	P	F1	P	R	F1	P	R	F1	P	R	F1	
LLaMa-3.3 (70B)	Baseline 1	52.8	50.0	51.4	31.3	37.7	34.2	29.6	40.4	34.2	28.4	31.8	30.0	32.8
	Baseline 2	65.4	63.0	64.2	26.7	**69.3**	38.5	17.0	56.8	26.2	11.2	22.5	15.0	26.6
	Newman et al.	61.9	60.0	60.9	36.4	40.5	38.3	32.8	44.5	37.8	29.5	30.2	29.8	35.3
	Ours	69.3	66.5	67.9	41.9	55.4	47.7	43.1	62.6	51.1	36.4	46.9	41.0	46.6
Mistral-Large (123B)	Baseline 1	54.7	51.5	53.0	33.1	34.5	33.8	31.6	30.4	31.0	15.5	24.7	19.0	27.9
	Baseline 2	66.8	64.0	65.4	27.4	65.0	38.5	22.7	47.4	30.7	17.8	30.7	22.6	30.6
	Newman et al.	67.9	65.5	66.7	39.9	41.6	40.7	34.7	46.3	39.7	29.9	35.1	32.3	37.6
	Ours	71.3	68.0	69.6	45.4	56.7	50.4	43.3	61.5	50.8	42.0	49.2	45.3	48.8
DeepSeek-V3 (685B)	Baseline 1	57.5	55.0	56.2	38.7	41.7	40.1	32.5	43.8	37.3	28.7	31.8	30.1	35.8
	Baseline 2	69.8	67.0	68.4	34.9	69.0	46.4	27.1	55.5	36.4	25.7	32.7	28.8	37.2
	Newman et al.	70.9	68.5	69.7	39.4	44.2	41.7	36.6	49.2	42.0	33.3	36.5	34.8	39.5
	Ours	74.3	71.0	72.6	39.6	56.9	46.7	47.7	65.2	55.1	40.4	49.8	44.6	48.8
GPT-4o-mini	Baseline 1	55.9	53.0	54.4	32.0	35.7	33.7	28.9	39.3	33.3	25.0	31.0	27.7	31.6
	Baseline 2	68.2	65.0	66.6	31.5	67.7	43.0	27.7	50.8	35.9	21.6	28.3	24.5	34.5
	Newman et al.	69.3	66.0	67.6	40.3	45.9	42.9	38.3	47.5	42.4	35.0	37.8	36.3	40.5
	Ours	72.6	69.0	70.8	46.5	59.7	52.3	**49.0**	**66.7**	**56.5**	43.5	51.9	47.3	52.0
GPT-4o	Baseline 1	58.5	56.0	57.2	35.8	43.2	39.2	36.9	41.8	39.2	29.0	34.7	31.6	36.7
	Baseline 2	70.2	67.0	68.6	34.2	68.0	45.5	27.9	56.0	37.2	19.4	33.6	24.6	35.8
	Newman et al.	71.3	68.5	69.9	45.0	47.9	46.4	38.7	49.8	43.6	36.9	40.0	38.4	42.8
	Newman et al. + COT	72.5	–	–	47.0	49.0	48.0	40.0	51.0	45.0	39.0	42.0	40.5	44.5
	Ours	**74.6**	**71.5**	**73.0**	51.5	59.4	55.2	46.1	66.7	54.5	45.9	55.7	50.3	53.3
	Ours + COT	75.8	–	–	53.0	60.5	56.5	48.0	68.0	56.3	47.0	57.0	51.5	55.0
GPT-o3	**Ours**	**77.2**	–	–	**55.0**	**62.0**	**58.3**	**50.0**	**70.0**	**58.3**	**49.0**	**59.0**	**53.5**	**56.7**

Table 2: Tabular evaluation results (%) on ARXIV2TABLE. All tasks use P/R/F1 as evaluation metrics. The best within each *backbone* is underlined and the best *overall* is **bold**.

tations of the papers for subsequent processing.

(B) Paper Batching Next, we divide all key information paragraphs into smaller batches. Processing too many papers at once negatively affects the model's performance (as demonstrated by the comparison of Baseline 1 in Table 2), whereas batching facilitates more efficient comparisons within each batch. For simplicity, we set a batch size of 4 and randomly partition R into $\left\lceil \frac{|R|}{4} \right\rceil$ batches.

(C) Paper Selection and Schema Refinement We initialize an empty schema and table, then sequentially process each batch with the LLM by providing it with the user's request and summaries of batched papers. The LLM is instructed to (1) decide whether each paper should be included or removed based on its key information and (2) refine the schema based on the current batch of papers. Schema refinement involves adding or removing specific columns or modifying existing values to align with different formats. For new papers that are deemed suitable for inclusion yet are not in the current table, we prompt the LLM to insert a new row with the extracted fields. This ensures that the table remains dynamically structured, continuously adapting to new information while maintaining consistency across batches.

Afterward, we iterate steps B and C for k iterations. Here k is a hyper-parameter and we set $k = 5$ in our experiments. The rationale is that multiple iterations allow the schema and table contents to progressively improve, ensuring better alignment with user demands. In each iteration, the batches are newly randomized so that each paper is compared with different subsets, enabling more robust decision-making and reducing bias from specific batch compositions. This iterative refinement also mitigates errors from earlier batches by revisiting and adjusting prior decisions based on newly processed information. After the final iteration, we re-verify each populated cell directly against the source text; unsupported values are set to N/A.

6 Experiments and Analyses

6.1 Experiment Setup

To demonstrate the generalizability of our method and evaluations, we conduct experiments using three proprietary and three open-source LLMs as backbone model representatives: GPT-4o (OpenAI, 2024b), GPT-4o-mini (OpenAI, 2024a), GPT-o3 (OpenAI, 2025b), DeepSeek-V3 (685B; DeepSeek-AI et al., 2024), LLaMa-3.3 (70B; Dubey et al., 2024), and Mistral-Large (123B; Mistral-AI, 2024). We apply all baseline methods and our proposed method to each model and use our evaluation framework to assess the quality of the generated tables based on our benchmark, focusing on four aspects: paper selection (**Paper**), schema content overlap (**Schema**), single-

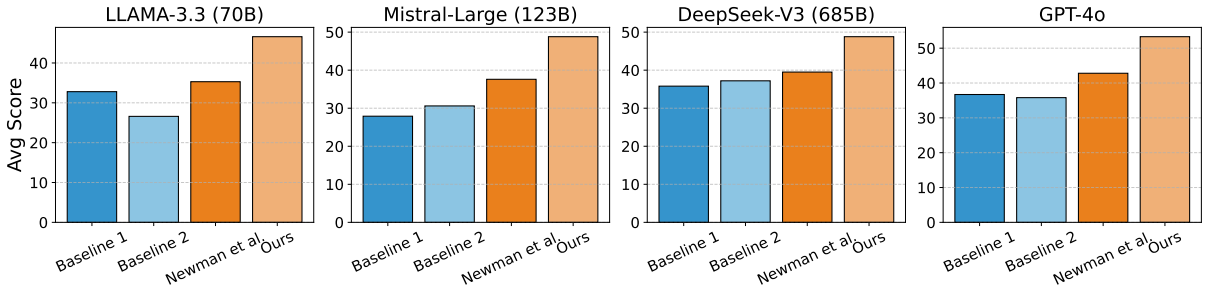

Figure 5: Average performance scores of four backbone LLMs across four different methods. The comparison highlights the consistent improvement of our proposed method over existing baselines and prior work.

cell value overlap (**Unary Value**), and comparisons across cells (**Pairwise Value**)[1]. For paper selection (T2) and all T3 dimensions, we report precision (P), recall (R), and F1. For the COT and GPT-o3 variants, T2 P/F1 are omitted on this slice due to compute limits. Due to cost, GPT-o3 is evaluated on a fixed subset.

6.2 Main Evaluation Results

We report the main results in Table 2 and summarize key findings below; a model-wise comparison across methods is shown in Figure 5.

(1) All methods and models struggle to distinguish relevant papers from distractors. Even at their best settings, LLaMa-3.3 and GPT-4o reach only 65.4–74.6% recall and 60.0–71.5% precision in T2 (Table 2), indicating substantial distractor inclusion or missed relevant papers depending on the operating point. We also find that processing papers individually, or using only abstracts for inclusion decisions, performs better than concatenating full texts, suggesting that overly long prompts can weaken per-paper inclusion decisions.

(2) Aligning generated schemas with the ground-truth table remains challenging. Among the baselines, the second method tends to achieve higher recall (e.g., 69.3% with LLaMa-3.3), largely because it produces more columns and thus overlaps more often with the ground-truth schema. Other methods have notably lower recall, indicating that generating meaningful columns that match the gold structure remains difficult.

(3) While unary values are well preserved, pairwise comparisons suffer substantial losses. Most methods, especially ours, achieve relatively strong unary F1 scores, whereas extracting and preserving pairwise relationships remains challenging. This trend holds across models: systems often identify individual entries correctly but fail to capture relations between them, highlighting the difficulty of preserving complex relational comparisons in generated tables.

(4) Our proposed method improves performance across all aspects and models. Across backbone models and evaluation criteria, our method consistently outperforms the baselines. It achieves the strongest overall results and the highest unary/pairwise F1, demonstrating robustness in both distractor handling and precise table generation.

(5) Larger models lead to better performance. For open-source LLMs, increasing model size consistently improves results under the same method. For GPT-4o, adding CoT yields consistent T3 gains with similar or slightly higher token budgets, while GPT-o3 attains the highest T3 scores overall, suggesting that stronger backbones better exploit iterative batching.

6.3 Validation of Utilization-Based Evaluation

To verify the reliability of synthesizing QA pairs using LLMs for evaluating tabular data, we conduct two complementary expert assessments. First, we invited the authors (as domain experts) to manually inspect a random sample of 200 QA pairs—spanning schema-level, unary value, and pairwise value comparisons. Annotators were asked to assess (1) whether each QA pair is firmly grounded in the source table, and (2) whether the LLM's answer is correct based on the generated target table. As shown in Table 3, the expert acceptance rates exceed 98% in all categories, confirming the quality of the synthesized QA pairs. Note that Table 4 measures evaluator–human agreement (reliability), not end-to-end table quality, so inter-method gaps are expected to be smaller than in Table 2.

[1]For paper selection, recall is emphasized instead of precision since missing relevant papers is more critical than including slightly noisy ones in literature review tasks.

Table	Schema	Unary Value	Pairwise Value
Source	99.5%	100%	98.5%
Target	98.5%	99.5%	97.0%

Table 3: Expert acceptance rate for the synthesized QA pairs sampled from our evaluations.

Method	LLM Rate	Human Rate	Agreement
Baseline 1	39.1%	39.6%	97.3%
Baseline 2	57.1%	57.3%	98.2%
Newman et al.	42.9%	43.0%	98.6%
Ours	57.3%	57.5%	98.0%

Table 4: Comparison between GPT-4o and human annotators on 300 QA pairs. We report the proportion of "yes" answers by each and their overall agreement.

Second, we conducted an additional human study to assess whether our LLM-based evaluation aligns with human judgment across different generation methods. For each method, we sampled 300 QA pairs, answered them using both LLMs and human annotators, and measured the agreement rate. As shown in Table 4, LLM and human "yes" response rates are highly consistent, with over 97% agreement across all methods. These results reinforce the robustness of our evaluation framework, demonstrating that LLM-synthesized QA pairs provide a scalable and trustworthy proxy for human judgment in assessing semantically diverse tabular outputs. Specifically, these results indicate that the high agreement is not driven by an inherent bias of LLMs toward their own generated QA pairs.

6.4 Batch Size and Iteration Sensitivity

We analyze the effect of batch size b and iteration count k in our iterative pipeline, which repeatedly performs paper selection and schema/table refinement over multiple batches. Using GPT-4o as the backbone, we evaluate 60 tables sampled to cover a range of sizes and schema complexity. We vary $b \in \{2, 4, 6\}$ and $k \in \{2, \ldots, 5\}$, and report T3 macro-F1, the average of Schema, Unary, and Pairwise F1. Each configuration is run with two seeds and averaged, with standard deviation shown where relevant. Token budgets follow the main-experiment prompt templates, and per-iteration context lengths remain below 128K.

Table 5 shows that performance improves steadily over the first few iterations, confirming the benefit of iterative refinement across different paper subsets. Gains saturate around $k \approx 4$–5, with b=4 performing slightly better overall. The best

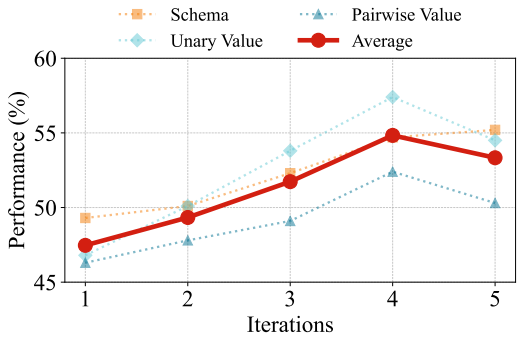

Figure 6: Ablation study on the number of iterations for our iterative batch-based table generation method.

macro-F1 is achieved at b=4, k=4 (53.1), while k=5 yields almost no further improvement. Across seeds, variability is modest (≤ 0.5 points).

k	Macro-F1 by batch size b=2	b=4	b=6	Std. dev. (avg) over seeds
2	50.1	50.7	50.5	0.5
3	51.8	52.2	52.0	0.4
4	52.6	**53.1**	52.9	0.4
5	**52.7**	53.0	**52.9**	0.4

Table 5: Macro-F1 across iterations k and batch sizes b. Gains saturate by $k \approx 4$–5, with b=4 slightly ahead.

Figure 6 shows the same trend across iterations. Most gains occur before iteration 4, after which performance plateaus. Later rounds may also introduce unsupported values that slightly reduce precision. We further observe that pairwise F1 benefits the most, suggesting that repeated cross-batch comparisons mainly improve relational consistency. Overall, these results support k=4 or k=5 as a practical default.

7 Conclusions

In this work, we introduce an improved literature review table generation task that incorporates distractor papers and replaces table captions with abstract user demands to better align with real-world scenarios, and curated an associated benchmark. Additionally, we propose an annotation-free evaluation framework using LLM-synthesized QA pairs and a novel method to enhance table generation. Our experiments show that current LLMs and existing methods struggle with our task, while our approach significantly improves performance. We envision that our work paves the way for more automated and scalable literature review table generation, ultimately facilitating the efficient synthesis of scientific knowledge in large-scale applications.

Limitations

A minor limitation is that our work uses ARXIVDIGESTABLES as the source of literature review tables for subsequent data reconstruction. However, Newman et al. (2024) have included their pipeline for scalably extracting literature review tables from scientific papers, thus resolving the data reliance gap (Ou et al., 2025; Gao et al., 2025). Beyond the computer science domain, our formulation and methodology are readily applicable to other scientific fields such as medicine, physics, and social sciences, where structured comparisons across publications are equally valuable. Moreover, the core task—generating structured tables from noisy, unstructured input with under-specified intent—extends naturally to real-world applications like news fact aggregation, personalized knowledge card generation, and structured database population from web or legal documents.

Another limitation of our work is its reliance on GPT-4o, a proprietary LLM, for benchmark curation and subsequent evaluation, which may introduce several issues. First, it raises concerns about data contamination (Deng et al., 2024a; Dong et al., 2024), as the model may generate user demands (during benchmark curation) and synthesis evaluation questions (when evaluating a generated table against the ground truth) that are similar to its training data, potentially leading to inflated performance in table generation. A data provenance check (Longpre et al., 2024) can be further implemented to address this issue. Second, the benchmark and evaluation process may inherit the internal knowledge or semantic distribution biases of GPT-4o, which could skew the evaluation of other LLMs and reduce the generalizability of our findings. Lastly, a minor issue is scalability, as curating larger datasets using a proprietary model can be resource-intensive and may limit accessibility when extending our framework to other literature or domains. Future work can explore the use of open-source LLMs to replicate the entire process for convenient adaptation to other tabular datasets.

Ethics Statement

The ARXIVDIGESTABLES (Newman et al., 2024) dataset used in our work is shared under the Open Data Commons License, which grants us access to it and allows us to improve and redistribute it for research purposes. Regarding language models, we access all open-source LMs via the Hugging Face Hub (Wolf et al., 2020) and proprietary GPT models through their official API[2]. The number of these models, if available, is marked in Table 2. All associated licenses for these models permit user access for research purposes, and we commit to following all terms of use.

When prompting GPT-4o to generate user demands and synthetic QA questions, we explicitly state in the prompt that the LLM should not generate any content that contains personal privacy violations, promotes violence, racial discrimination, hate speech, sexual, or self-harm contents. We also manually inspect a random sample of 100 data entries generated by GPT-4o for offensive content, and none are detected. Therefore, we believe that our dataset is safe and will not yield any negative or harmful impact.

Our human annotations are conducted by trained graduate-level annotators (n=19), compensated above local minimum wage. They have sufficient experience in data collection for training large language models and are proficient in English, primarily from Asia. They receive thorough training on the task and are reminded to have a clear understanding of the task instructions before proceeding to annotation. The high level of inter-agreement also confirms the quality of our annotation. The expert annotators have agreed to participate as their contribution to the paper without receiving any compensation.

Acknowledgments

We thank Muhan Gao and the JHU CLSP community for their discussions and inspiration, and the HKUST KnowComp community for their help with data annotation. The authors of this paper were supported by the ITSP Platform Research Project (ITS/189/23FP) from the Innovation and Technology Commission of Hong Kong SAR, China; the AoE (AoE/E-601/24-N), RIF (R6021-20), and GRF (16205322) from the Research Grants Council of Hong Kong SAR, China; and the ONR grant (N0001424-1-2089) from the U.S. Office of Naval Research.

References

Ojas Ahuja, Jiacheng Xu, Akshay Gupta, Kevin Horecka, and Greg Durrett. 2022. ASPECTNEWS: aspect-oriented summarization of news documents. In *Proceedings of the 60th Annual Meeting of the*

[2] https://platform.openai.com/

Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022, pages 6494–6506. Association for Computational Linguistics.

John Dagdelen, Alexander Dunn, Sanghoon Lee, Nicholas Walker, Andrew S Rosen, Gerbrand Ceder, Kristin A Persson, and Anubhav Jain. 2024. Structured information extraction from scientific text with large language models. *Nature Communications*, 15(1):1418.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, and Matt Gardner. 2021. A dataset of information-seeking questions and answers anchored in research papers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 4599–4610. Association for Computational Linguistics.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, and Wangding Zeng. 2024. Deepseek-v3 technical report. *CoRR*, abs/2412.19437.

Chunyuan Deng, Yilun Zhao, Xiangru Tang, Mark Gerstein, and Arman Cohan. 2024a. Investigating data contamination in modern benchmarks for large language models. In *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*, pages 8706–8719. Association for Computational Linguistics.

Zheye Deng, Chunkit Chan, Weiqi Wang, Yuxi Sun, Wei Fan, Tianshi Zheng, Yauwai Yim, and Yangqiu Song. 2024b. Text-tuple-table: Towards information integration in text-to-table generation via global tuple extraction. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*, pages 9300–9322. Association for Computational Linguistics.

Jay DeYoung, Iz Beltagy, Madeleine van Zuylen, Bailey Kuehl, and Lucy Lu Wang. 2021. Ms\^2: Multi-document summarization of medical studies. In *Proceedings of the 2021 Conference on Empirical Meth-*

ods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021, pages 7494–7513. Association for Computational Linguistics.

Yihong Dong, Xue Jiang, Huanyu Liu, Zhi Jin, Bin Gu, Mengfei Yang, and Ge Li. 2024. Generalization or memorization: Data contamination and trustworthy evaluation for large language models. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 12039–12050. Association for Computational Linguistics.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Grégoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, and et al. 2024. The llama 3 herd of models. *CoRR*, abs/2407.21783.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Muhan Gao, Jash Shah, Weiqi Wang, and Daniel Khashabi. 2025. Science hierarchography: Hierarchical organization of science literature. *CoRR*, abs/2504.13834.

Hayato Hashimoto, Kazutoshi Shinoda, Hikaru Yokono, and Akiko Aizawa. 2017. Automatic generation of review matrices as multi-document summarization of scientific papers. In *Proceedings of the 2nd Joint Workshop on Bibliometric-enhanced Information Retrieval and Natural Language Processing for Digital Libraries (BIRNDL 2017) co-located with the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2017), Tokyo, Japan, August 11, 2017*, volume 1888 of *CEUR Workshop Proceedings*, pages 69–82. CEUR-WS.org.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur P. Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. *Trans. Assoc. Comput. Linguistics*, 7:452–466.

Yoonjoo Lee, Kyungjae Lee, Sunghyun Park, Dasol Hwang, Jaehyeon Kim, Hong-In Lee, and Moontae Lee. 2023. QASA: advanced question answering on scientific articles. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 19036–19052. PMLR.

Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, Ming Zhou, and Zhoujun Li. 2020. Tablebank: Table benchmark for image-based table detection and recognition. In *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020*, pages 1918–1925. European Language Resources Association.

Tong Li, Zhihao Wang, Liangying Shao, Xuling Zheng, Xiaoli Wang, and Jinsong Su. 2023. A sequence-to-sequence&set model for text-to-table generation. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 5358–5370. Association for Computational Linguistics.

Shayne Longpre, Robert Mahari, Naana Obeng-Marnu, William Brannon, Tobin South, Katy Ilonka Gero, Alex Pentland, and Jad Kabbara. 2024. Position: Data authenticity, consent, & provenance for AI are all broken: what will it take to fix them? In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Xinyuan Lu, Liangming Pan, Qian Liu, Preslav Nakov, and Min-Yen Kan. 2023. SCITAB: A challenging benchmark for compositional reasoning and claim verification on scientific tables. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 7787–7813. Association for Computational Linguistics.

Yao Lu, Yue Dong, and Laurent Charlin. 2020. Multi-xscience: A large-scale dataset for extreme multi-document summarization of scientific articles. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 8068–8074. Association for Computational Linguistics.

Mistral-AI. 2024. Large enough. *Mistral AI Blog*.

Nafise Sadat Moosavi, Andreas Rücklé, Dan Roth, and Iryna Gurevych. 2021. Scigen: a dataset for reasoning-aware text generation from scientific tables. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*.

Benjamin Newman, Yoonjoo Lee, Aakanksha Naik, Pao Siangliulue, Raymond Fok, Juho Kim, Daniel S. Weld, Joseph Chee Chang, and Kyle Lo. 2024. Arxivdigestables: Synthesizing scientific literature into tables using language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*, pages 9612–9631. Association for Computational Linguistics.

OpenAI. 2024a. Gpt-4o mini: advancing cost-efficient intelligence. *OpenAI*.

OpenAI. 2024b. Hello gpt-4o. *OpenAI*.

OpenAI. 2025a. Introducing deep research. *OpenAI Blog*.

OpenAI. 2025b. Introducing openai o3 and o4-mini. *OpenAI*.

OpenAI. 2025c. Openai o3-mini. *OpenAI Blog*.

Jiefu Ou, William Gantt Walden, Kate Sanders, Zhengping Jiang, Kaiser Sun, Jeffrey Cheng, William Jurayj, Miriam Wanner, Shaobo Liang, Candice Morgan, Seunghoon Han, Weiqi Wang, Chandler May, Hannah Recknor, Daniel Khashabi, and Benjamin Van Durme. 2025. CLAIMCHECK: how grounded are LLM critiques of scientific papers? In *Findings of the Association for Computational Linguistics: EMNLP 2025, Suzhou, China, November 4-9, 2025*, pages 21712–21735. Association for Computational Linguistics.

Vishakh Padmakumar, Joseph Chee Chang, Kyle Lo, Doug Downey, and Aakanksha Naik. 2025. Setting the table with intent: Intent-aware schema generation and editing for literature review tables. To appear.

Pritika Ramu, Aparna Garimella, and Sambaran Bandyopadhyay. 2024. Is this a bad table? A closer look at the evaluation of table generation from text. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*, pages 22206–22216. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3980–3990. Association for Computational Linguistics.

Daniel M. Russell, Mark Stefik, Peter Pirolli, and Stuart K. Card. 1993. The cost structure of sensemaking. In *Human-Computer Interaction, INTERACT '93, IFIP TC13 International Conference on Human-Computer Interaction, 24-29 April 1993, Amsterdam, The Netherlands, jointly organised with ACM Conference on Human Aspects in Computing Systems CHI'93*, pages 269–276. ACM.

Liangtai Sun, Yang Han, Zihan Zhao, Da Ma, Zhennan Shen, Baocai Chen, Lu Chen, and Kai Yu. 2024. Scieval: A multi-level large language model evaluation benchmark for scientific research. In *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pages 19053–19061. AAAI Press.

Anirudh Sundar, Christopher Richardson, and Larry Heck. 2024. gtbls: Generating tables from text by conditional question answering. *CoRR*, abs/2403.14457.

Xiangru Tang, Yiming Zong, Jason Phang, Yilun Zhao, Wangchunshu Zhou, Arman Cohan, and Mark Gerstein. 2023. Struc-bench: Are large language models really good at generating complex structured data? *CoRR*, abs/2309.08963.

Qwen Team. 2025. Qwen3 technical report. *CoRR*, abs/2505.09388.

Weiqi Wang, Limeng Cui, Xin Liu, Sreyashi Nag, Wenju Xu, Chen Luo, Sheikh Muhammad Sarwar, Yang Li, Hansu Gu, Hui Liu, Changlong Yu, Jiaxin Bai, Yifan Gao, Haiyang Zhang, Qi He, Shuiwang Ji, and Yangqiu Song. 2025a. Ecomscriptbench: A multi-task benchmark for e-commerce script planning via step-wise intention-driven product association. In *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*, pages 1–22. Association for Computational Linguistics.

Weiqi Wang, Tianqing Fang, Wenxuan Ding, Baixuan Xu, Xin Liu, Yangqiu Song, and Antoine Bosselut. 2023a. CAR: conceptualization-augmented reasoner for zero-shot commonsense question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, Findings of ACL, pages 13520–13545. Association for Computational Linguistics.

Weiqi Wang, Tianqing Fang, Chunyang Li, Haochen Shi, Wenxuan Ding, Baixuan Xu, Zhaowei Wang, Jiaxin Bai, Xin Liu, Cheng Jiayang, Chunkit Chan, and Yangqiu Song. 2024a. CANDLE: iterative conceptualization and instantiation distillation from large language models for commonsense reasoning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1:*

Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024, pages 2351–2374. Association for Computational Linguistics.

Weiqi Wang, Tianqing Fang, Haochen Shi, Baixuan Xu, Wenxuan Ding, Liyu Zhang, Wei Fan, Jiaxin Bai, Haoran Li, Xin Liu, and Yangqiu Song. 2025b. On the role of entity and event level conceptualization in generalizable reasoning: A survey of tasks, methods, applications, and future directions. In *Findings of the Association for Computational Linguistics: EMNLP 2025, Suzhou, China, November 4-9, 2025*, pages 2260–2281. Association for Computational Linguistics.

Weiqi Wang, Tianqing Fang, Baixuan Xu, Chun Yi Louis Bo, Yangqiu Song, and Lei Chen. 2023b. CAT: A contextualized conceptualization and instantiation framework for commonsense reasoning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 13111–13140. Association for Computational Linguistics.

Weiqi Wang, Xin Liu, Binxuan Huang, Hejie Cui, Rongzhi Zhang, Changlong Yu, Shuowei Jin, Jingfeng Yang, Qingyu Yin, Zhengyang Wang, Zheng Li, Yifan Gao, Priyanka Nigam, Bing Yin, Lihong Li, and Yangqiu Song. 2026. Heapa: Difficulty-aware heap sampling and on-policy query augmentation for LLM reinforcement learning. *CoRR*, abs/2601.22448.

Weiqi Wang and Yangqiu Song. 2025. MARS: benchmarking the metaphysical reasoning abilities of language models with a multi-task evaluation dataset. In *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*, pages 1568–1596. Association for Computational Linguistics.

Xingbo Wang, Samantha L. Huey, Rui Sheng, Saurabh Mehta, and Fei Wang. 2024b. Scidasynth: Interactive structured knowledge extraction and synthesis from scientific literature with large language model. *CoRR*, abs/2404.13765.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2020 - Demos, Online, November 16-20, 2020*, pages 38–45. Association for Computational Linguistics.

Xueqing Wu, Jiacheng Zhang, and Hang Li. 2022. Text-to-table: A new way of information extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 2518–2533. Association for Computational Linguistics.

Shuo Zhang and Krisztian Balog. 2018. On-the-fly table generation. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, pages 595–604. ACM.

Appendices

A Additional Analysis

A.1 Method Efficiency Evaluations

In addition to quality metrics (Table 2), we compare generation efficiency based on (i) generation success rate (GSR) under the backbone context window limit, (ii) average token usage per table, and (iii) average runtime. These statistics are reported in Table 9. Overall, our method achieves a 100% success rate while keeping token usage controlled, indicating that iterative batching improves robustness without incurring excessive context overhead.

To assess the efficiency and scalability of our iterative batch-based method, we report computational statistics in Table 9. Each method was run using the same LLaMA-3.3 model backend. We measure three aspects: (1) generation success rate, defined as the proportion of prompts yielding complete tables within the context window, (2) average token usage per table, and (3) average runtime per table. Our method achieves a 100% success rate, outperforming the baselines that occasionally fail due to context limitations or prompt instability. While our runtime is moderately longer than Baseline 1 and Baseline 2, it remains comparable to Newman et al. and stays well within acceptable latency for practical usage. Furthermore, token usage remains controlled, confirming that our iterative approach does not incur excessive computational cost despite its multi-step structure. These results demonstrate that our method offers a favorable trade-off between performance and efficiency.

A.2 Additional Backbone Comparisons (GPT-5.1 and GPT-o3)

To test whether the relative improvements of our iterative batch-based method persist under stronger proprietary backbones, we run additional comparisons with GPT-5.1 and GPT-o3 under the same evaluation protocol as in the main experiments. Due to cost constraints, GPT-o3 results are reported on a fixed 50-table subset. We report T3 utilization metrics (Paper/Schema/Unary/Pairwise F1 and their average) in Table 6.

A.3 Evaluator Independence Checks

This section tests whether our utilization-based evaluation depends on which LLM synthesizes QA pairs versus which LLM answers them. We decouple *QA synthesis* and *answering*: a synthesizer produces all schema/unary/pairwise QAs from a table; a distinct answerer then answers those QAs from the comparison table. We then swap roles and recompute scores. Stability is measured by (i) rank correlations of method orderings (Spearman, Kendall) over model×method tuples, and (ii) the maximum absolute F1 change per dimension (Schema/Unary/Pairwise).

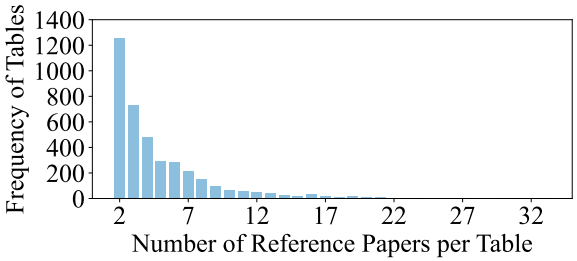

Figure 7: Distribution of number of papers in each table.

Setup. We evaluate four synthesizer→answerer pairs: GPT-4o→GPT-4o-mini, GPT-4o-mini→GPT-4o, GPT-4o→Llama-3.3, Llama-3.3→GPT-4o (Table 11), and additionally include a cross-family swap using `Qwen3-30B-A3B-Instruct` (Table 12). We randomly sample 50 tables (stratified by table size) and include all generation methods (Baseline 1, Baseline 2, Newman et al., Ours). Decoding temperature is 0.5; the same seeds as the main runs are reused. Each run uses the same QA quantity as the main evaluation (all schema and unary QAs, 10 pairwise QAs per table).

Result. Across swaps, macro Kendall $\approx$ 0.68–0.71 and macro Spearman $\approx$ 0.84–0.87, while max absolute F1 changes remain $\leq$ 2.1 points. Method rankings are therefore stable, and absolute performance differences are small, suggesting limited evaluator dependence and addressing circularity concerns.

Cross-family evaluator swap with Qwen3. In addition to the swaps above, we also test evaluator independence by replacing GPT-4o with the open-source `Qwen3-30B-A3B-Instruct` (Team, 2025) as either the QA synthesizer or the answerer on a subset. We report the relative change in macro-F1 (vs. GPT-4o as the default evaluator) and ranking stability in Table 12.

Together with Tables 10–11, these results indicate that method rankings are stable under evaluator changes across both proprietary and open-source model families.

Backbone & Method	Paper F1	Schema F1	Unary F1	Pairwise F1	Avg
GPT-5.1 + Baseline 2	76.0	55.0	49.0	38.0	47.3
GPT-5.1 + Ours	78.2	59.3	59.3	54.5	57.8
GPT-o3 + Baseline 2 (50-table subset)	75.0	53.0	47.0	36.0	45.3
GPT-o3 + Ours (50-table subset / full set)	77.2	58.3	58.3	53.5	56.7

Table 6: Results under stronger proprietary backbones using the same utilization-based evaluation protocol. GPT-o3 rows are evaluated on a fixed subset due to cost; GPT-5.1 rows follow the same evaluation setup as the corresponding main experiments.

Method	GSR	#Tokens
Baseline 1	48.19%	128K
Baseline 2	98.23%	167K
Newman et al.	99.71%	110K
Ours	100.0%	118K

Table 7: Comparison of the efficiency of different methods. GSR stands for generation success rate.

Statistic	Paper Count	Column Count	Distractor Count
Min	1	2	4
Max	32	13	10
Mean	3.65	3.56	5.21
Total	7158	6967	10196

Table 8: Summary statistics of the ARXIV2TABLE benchmark. We report aggregate values for the number of papers, columns, and distractor papers per table.

A.4 Additional Retrieval Baselines and Rationale

We compare dense and sparse retrieval for constructing candidate pools used in distractor verification. Each table's user demand is matched against all paper titles and abstracts. We evaluate TF–IDF, BM25, and a dense retriever based on `sentence-t5-xxl` (SentenceTransformers) on a 200-table slice. All methods return top-k candidates; we report macro Precision@k (P@k), Recall@k (R@k), and Average Precision (AP). Sparse methods use standard tokenization with stopword removal; dense retrieval encodes the concatenation of title and abstract and ranks by cosine similarity.

We additionally probe different cutoffs to assess stability. Results at smaller and larger k show the same ordering, with the dense method maintaining a recall advantage as k increases.

These results justify our use of an embedding-based retriever to construct a semantically challenging yet realistic candidate pool. Sparse baselines remain competitive in precision at very small cutoffs (e.g., @5), but they under-recall paraphrastic matches common in scientific text. The dense approach reduces downstream risk of missing truly relevant papers during human verification.

Method	Success Rate	#Tokens	Avg. Runtime
Baseline 1	48.2%	128K	37
Baseline 2	98.2%	167K	118
Newman et al.	99.7%	110K	208
Ours	100.0%	118K	194

Table 9: Computational cost and efficiency metrics across different generation methods using LLaMA-3.3. We report the generation success rate, average token usage, and average runtime (s) per table.

B Implementation Details

In this section, we provide additional implementation details about our benchmark curation and evaluation pipeline, including the prompt we used and the models we accessed.

B.1 Prompts Used

We provide all prompts used in our benchmark construction and evaluation. In benchmark construction, the key steps are (i) rewriting captions into user demands without leaking schema/values and (ii) synthesizing QA pairs from the ground-truth table. In evaluation, we normalize tables, canonicalize headers, decouple QA synthesis from answering, and run a reverse-QA pass for precision.

User demand rewrite (benchmark construction). This prompt rewrites terse/arXiv-style captions into contextually self-contained user demands that specify the table's purpose without leaking column names or specific values. It improves realism by emulating well-specified but abstract requests while keeping evaluation fair.

```
Given a literature review table, along
with its caption, you are tasked with
writing a user demand or intention for
the creator of this table. The user
demand should be written as though
you are instructing an AI system to
```

Synth→Ans	Schema F1		Unary F1		Pairwise F1	
	Mean Δ	Max Δ	Mean Δ	Max Δ	Mean Δ	Max Δ
4o → 4o-mini	0.4	1.5	0.5	1.6	0.6	1.8
4o-mini → 4o	0.5	1.7	0.6	1.8	0.7	1.9
4o → Llama3.3	0.6	1.9	0.7	2.0	0.9	2.1
Llama3.3 → 4o	0.5	1.7	0.6	1.9	0.8	2.0

Table 10: Absolute F1 drift (percentage points) when swapping the QA synthesizer and answerer. Values are averaged over model×method tuples.

Synth→Ans	Spearman (Schema/Unary/Pairwise) ↑	Kendall (Schema/Unary/Pairwise) ↑	Macro Spearman ↑	Macro Kendall ↑
4o → 4o-mini	0.87 / 0.88 / 0.86	0.71 / 0.72 / 0.70	0.87	0.71
4o-mini → 4o	0.85 / 0.86 / 0.86	0.69 / 0.70 / 0.70	0.86	0.70
4o → Llama3.3	0.83 / 0.84 / 0.85	0.67 / 0.68 / 0.69	0.84	0.68
Llama3.3 → 4o	0.84 / 0.85 / 0.86	0.68 / 0.69 / 0.70	0.85	0.69

Table 11: Ranking stability across methods under evaluator swaps. Correlations computed over model×method tuples.

```
generate the table. Avoid directly mentioning column names in the table itself, but instead, focus on explaining why the table is needed and what information it should contain. You may include a description of the table's structure, whether it requires detailed or summarized columns. Additionally, infer the user's intentions from the titles of the papers the table will include. Limit each user demand to 1-2 sentences. Examples of good user demands are: I need a table that outlines how each study conceptualizes the problem, categorizes the task, describes the data analyzed, and summarizes the main findings. The table should have detailed columns for each of these aspects. Generate a detailed table comparing the theoretical background, research methodology, and key results of these papers. You can use several columns to capture these aspects for each paper. I want to create a table that summarizes the datasets used to evaluate different GNN models, focusing on the common features and characteristics found across the papers listed below. The table should have concise columns to highlight these dataset attributes. Now, write a user demand for the table below. The caption of the table is "<CAPTION>". The table looks like this:
<TABLE>
The following papers are included in the table:
<PAPER-1> . . . <PAPER-N>
Write the user demand for this table. Do not include the column names in the user demand. Write a concise and clear user demand covering the function, topic, and structure of the table with one or two sentences. The user demand is:
```

Leak check for user demands (guardrail). Immediately after rewriting, we run a leak check to ensure the demand does not expose schema labels or table values, and to auto-rewrite if needed. This keeps construction oracle-free and reproducible.

```
You are given: (i) a user demand written from a caption, and (ii) the target table (schema and a few cell values).
Decide if the user demand leaks any column names or specific cell values, or directly paraphrases them.
Return {ACCEPT, REWRITE} and a one-sentence reason. If REWRITE, produce a version that removes leaked tokens while staying specific and self-contained.
Output JSON only: {"decision": "...", "reason": "...", "demand": "..." }
```

QA synthesis from ground truth (recall side). This prompt generates binary QA pairs from the gold table to test whether a system table retains schema, values, and pairwise relations. It provides full coverage for schema/unary and a sampled, diverse set for pairwise.

Synthesizer & Answerer	Relative ΔF1 vs. GPT-4o	Spearman ρ	Kendall τ
GPT-4o → Qwen3-30B-A3B	+1.7	0.89	0.78
Qwen3-30B-A3B → GPT-4o	+2.4	0.86	0.74

Table 12: Cross-family evaluator swap using `Qwen3-30B-A3B-Instruct`. Relative ΔF1 is computed against the default GPT-4o-based evaluation on the same subset; rank correlations are computed over model×method tuples.

Constraint	**Operationalization (what we enforce)**
Self-contained	Mentions the topic and the intended comparison goal without assuming access to the gold table.
Goal-oriented	Specifies what decision/analysis the table should support (e.g., comparing families of methods, tracing trends, or summarizing benchmark settings).
Non-leaking	Avoids verbatim column headers and specific numeric/string values from the gold table; avoids directly paraphrasing headers.
Concise	1–2 sentences; avoids long enumerations of fields to keep the request natural.

Table 13: Checklist used for caption-to-demand rewriting. The automatic leak check (Appendix B) flags violations and triggers rewriting.

Method	P@10	R@10	AP
TF–IDF	0.44	0.36	0.31
BM25	0.49	0.40	0.36
SentenceBERT	**0.54**	**0.47**	**0.42**

Table 14: Top-10 retrieval quality (macro). Dense retrieval improves recall and AP, which is critical for minimizing false negatives in later filtering.

```
You will evaluate the quality of a generated table by comparing it against a ground-truth table. The goal is to assess whether the generated table correctly retains the schema, individual values, and pairwise relationships. This is achieved by generating targeted QA pairs based on the ground-truth table and answering them using the generated table. Step 1: QA Pair Generation Based on the Ground-Truth Table Generate binary (Yes/No) QA pairs focusing on three aspects: Schema QA Pairs: Check whether a specific column from the ground-truth table appears in the generated table schema. Example: Is Dataset included in the table schema? Unary Value QA Pairs: Check whether a specific cell value from the ground-truth table is present in the generated table. Example: Is
```

Method	@5		@20	
	P@5	R@5	P@20	R@20
TF–IDF	0.58	0.25	0.35	0.49
BM25	0.62	0.28	0.38	0.55
SentenceBERT	**0.65**	**0.33**	**0.40**	**0.62**

Table 15: Precision/recall at @5 and @20. Dense retrieval sustains higher recall across cutoffs.

```
CL, TL the loss function for paper CN-LexNet? Pairwise Value QA Pairs: Check whether a relationship between two values remains consistent in the generated table. Example: Is ResNet-v2 using more evaluation metrics than GAN? For Schema and Unary Value, generate a QA pair for every column and every cell, respectively. For Pairwise Value, randomly sample 10 pairs per table and construct the corresponding QA pairs. Step 2: Answering QA Pairs Using the Generated Table After generating the QA pairs, answer them using the generated table. Provide only "yes" or "no" responses: If the information is present in the generated table, respond with "yes." If the information is missing or different, respond with "no." Your task is to generate the QA pairs based on the ground-truth table and then answer them based on the generated table. Now, begin by generating the QA pairs.
```

Answering gold QAs on the system table (decoupled answerer). When we decouple synthesis from answering (for evaluator-independence checks and robustness), we use an answer-only prompt that strictly binds answers to the system table content.

```
Input: (i) a normalized generated table, and (ii) a list of binary Yes/No QAs synthesized from the ground-truth table. Answer each QA using only the generated table. If explicitly supported and consistent, answer "yes"; otherwise "no".
```

```
Return one lowercase token per line:
yes/no (no explanations).
```

Reverse-QA synthesis from the system table (precision side). To measure precision, we synthesize QAs from the system table and answer them on the gold table. This credits only content supported by the gold table and rejects unsupported "novel" entries.

```
Given a normalized generated table,
create binary Yes/No QAs for: (i)
schema presence, (ii) specific cell
values, and (iii) pairwise relations
(10 pairs).
Write unambiguous questions that can
be answered solely from this generated
table. Avoid trivial or duplicate QAs.
Return JSON list:
["type":"schema|unary|pairwise","q":"..."
...]
```

Answering reverse-QAs on the gold table (precision side). This answer-only prompt binds answers to the gold table. By instruction, any system-only content is answered "no".

```
Input: (i) a normalized gold table, and
(ii) a list of Yes/No QAs synthesized
from the generated table.
Answer each QA using only the gold table.
Novel content not present in gold must
be answered "no".
Return one lowercase token per line:
yes/no.
```

QA validation and deduplication (quality filter). Before answering, we filter QAs to remove ill-formed, non-binary, trivial, or duplicate items. This reduces evaluator brittleness and ensures consistent scoring.

```
Given a list of binary QAs, remove
any that are ill-formed, non-binary,
trivially true/false, or duplicates.
Return the filtered list as JSON in the
same schema and include a "removed" list
with reasons.
```

Table normalization (robust parsing). We normalize both gold and system tables to a rectangular CSV with consistent headers and "N/A" for missing entries so that downstream prompts and scripts operate deterministically.

```
You are given a table in arbitrary
Markdown/LaTeX/CSV. Normalize it into a
rectangular CSV with a header row, one
row per paper.
- Trim whitespace; collapse multi-line
cells; preserve units and text verbatim;
fill missing cells with "N/A".
- Do not infer new columns or values.
Return CSV only (no commentary).
```

Schema alias canonicalization (header mapping). We canonicalize header synonyms (e.g., "Dataset" vs "Data") before computing schema scores to reduce false mismatches due to phrasing.

```
Given two header lists (gold vs system),
produce a one-to-one or one-to-many
mapping of system headers to gold
headers when they are semantically
equivalent.
Rules: prefer exact matches; allow
synonyms; keep units consistent; do not
map if semantics differ.
Return JSON: ["system": "...", "gold":
"...", "justification": "..." ...]
```

The distribution of number of papers per table in ARXIV2TABLE is shown in Figure 7.

B.2 Evaluation Implementations

We access all open-source LLMs via the Hugging Face library (Wolf et al., 2020). The models used are `meta-llama/Llama-3.3-70B-Instruct`, `mistralai/Mistral-Large-Instruct-2411`, and `deepseek-ai/DeepSeek-V3`. For GPT models, we access them via the official OpenAI Batch API[3]. The models used are `gpt-4o-mini-2024-07-18` and `gpt-4o-2024-08-06`. Note that the DeepSeek model family has a context window limit of 64K tokens, whereas the others have a limit of 128K tokens. The generation temperature is set to 0.5 for all experiments. All experiments are repeated twice and the average performance is reported.

We normalize both gold and generated tables to CSV grids with a single header row and "N/A" for missing cells, then apply schema-alias canonicalization before scoring. For the recall side, we synthesize QAs from the gold table and answer them on the system table; for the precision side, we synthesize QAs from the system table and answer them

[3] https://platform.openai.com/docs/guides/batch

on the gold table. Answers are strictly "yes"/"no"; we lowercase, strip punctuation, and parse the first token if extra text is emitted. Schema and unary QAs are exhaustive; for pairwise, we sample 10 pairs per table while avoiding duplicates and promoting coverage over distinct columns. Randomization uses fixed seeds when supported; otherwise we repeat evaluation twice and average.

For cross-evaluator checks, we decouple QA synthesis and answering and swap the evaluator models as reported in Appendix A.3. Decoding parameters use temperature 0.5, top_p 1.0, no repetition penalties, and a max_new_tokens budget that comfortably covers the QA list; timeouts are 120s per call with up to two retries on transient errors. Significance testing for main-table comparisons uses table-level bootstrap (10,000 resamples) with 95% confidence intervals; full CIs and prompt JSON I/O schemas will be released with code. For the additional cross-family evaluator swap, we also use `Qwen3-30B-A3B-Instruct` under the same decoding settings (Wang et al., 2026, 2024a, 2025b, 2023b,a, 2025a; Wang and Song, 2025).

C Annotation Details

To ensure high-quality annotations, we recruited 19 trained graduate-level annotators with computer-science research experience and admitted them only after passing qualification rounds. For each candidate paper, annotators received clear, layman-friendly instructions with definitions and multiple examples, then confirmed they had read them via a checkbox before starting. Using the interface in Figure 8, each instance was judged with a binary {include, exclude} decision relative to the user demand and the known reference papers (title+abstract basis). Every instance received two independent labels; on disagreement, two additional adjudicators resolved to consensus. We continuously monitored performance, provided targeted feedback on common errors, and removed spammers or underperformers. The study ran for eleven days; first-pass agreement reached 94% pairwise with Fleiss' κ=0.73, the self-reported median time per decision was about seven minutes, and annotators were compensated above the local minimum wage. The final corpus contains 10,196 curated distractor labels across 1,957 tables (with 10 initial candidates per table before filtering), supporting the quality reported in Section 4.2.

D Case Studies

Table 16 shows examples of original captions and rewritten user demands, illustrating how short or context-dependent captions can be transformed into self-contained requests that better specify the intended comparison goal. Table 17 provides illustrative examples of schema, unary, and pairwise questions used by our utilization-based evaluation to measure schema coverage, factual retention, and relational consistency.

We additionally include two example pairs of ground-truth and generated tables in Table 18. These examples highlight common behaviors in literature-review table generation: systems can enrich tables by adding helpful organizing fields, but may also omit important attributes from the ground truth or introduce overly fine-grained fields that dilute the central comparison. Overall, the examples underscore the importance of jointly handling paper selection, schema induction, and value grounding.

Original Table Caption	User Demand
Comparison of trajectory and path planning approaches	I need a table that compares recent trajectory/path planning methods, emphasizing how they handle safety constraints (e.g., collision avoidance) and what assumptions they make about the environment. Please summarize the key strengths, limitations, and typical application settings for each approach.
Publications with deep-learning focused sampling methods. We cluster the papers based on the space the sample through and how the samples are evaluated. Some approaches further consider an optional refinement stage.	Please create a table that organizes learning-based sampling methods by how candidates are generated and how they are scored, so it is easy to see the main design choices across papers. If a method uses an extra refinement step or additional supervision, highlight that in the comparison.
Categorization of textual explanation methods.	I want a table that groups approaches for producing textual explanations by what kind of explanation they generate and what evidence they rely on (e.g., rationales, templates, retrieved facts). The goal is to quickly compare how different families of methods justify their predictions.
Metadata of the three benchmarks that we focus on. XSumSota is a combined benchmark of cite:1400aac and cite:d420ef8 for summaries generated by the state-of-the-art summarization models.	Please produce a table that contrasts these benchmarks in terms of what they measure and how they are evaluated, including how labels are obtained and what the evaluation protocol looks like. I want to understand which benchmark is most suitable for comparing modern summarization systems.
Review of open access ground-based forest datasets	I need a table summarizing open-access forest datasets, focusing on what is recorded, when and where data are collected, and what tasks each dataset supports. The table should make it easy to choose a dataset for a specific forest monitoring objective.
Comparison of existing consistency-type models.	Please construct a table comparing consistency-oriented models by what notion of consistency they enforce and how they operationalize it (objective, constraints, or training signal). The table should make clear how each model differs in assumptions and what settings it is intended for.

Table 16: Examples of original captions and rewritten user demands. User demands are written to be self-contained and goal-oriented while avoiding direct leakage of gold headers or specific cell values.

Schema QA (presence of a field)	Unary QA (a specific entry)	Pairwise QA (a relation)
Is the evaluation benchmark included in the table schema?	Is `ZsRE` listed as an evaluation benchmark for `ROME`?	Does `ROME` achieve higher reported accuracy than `MEND` on `ZsRE`?
Is the training signal/objective described in the table schema?	Is `consistency regularization` listed as a training signal for at least one method in the table?	Do methods that use `consistency regularization` report better performance than those that do not on the same benchmark?
Is the data source or dataset type included in the table schema?	Is `ImageNet` listed as a dataset used in any compared method?	Is `Method A` evaluated on more datasets than `Method B` in the table?
Is the compute setting (e.g., hardware or budget) included in the table schema?	Is `8×A100` listed as the hardware setting for any method?	Does `Method A` report a lower compute budget than `Method B` under the same evaluation setup?
Is the publication year or timeframe included in the table schema?	Is `2023` listed as the publication year for `Method X`?	Are post-2022 methods reported as outperforming pre-2020 methods on the same metric in the table?

Table 17: Illustrative examples of schema/unary/pairwise questions. In the benchmark, questions are synthesized from each ground-truth table and are answerable by reading the corresponding comparison table.

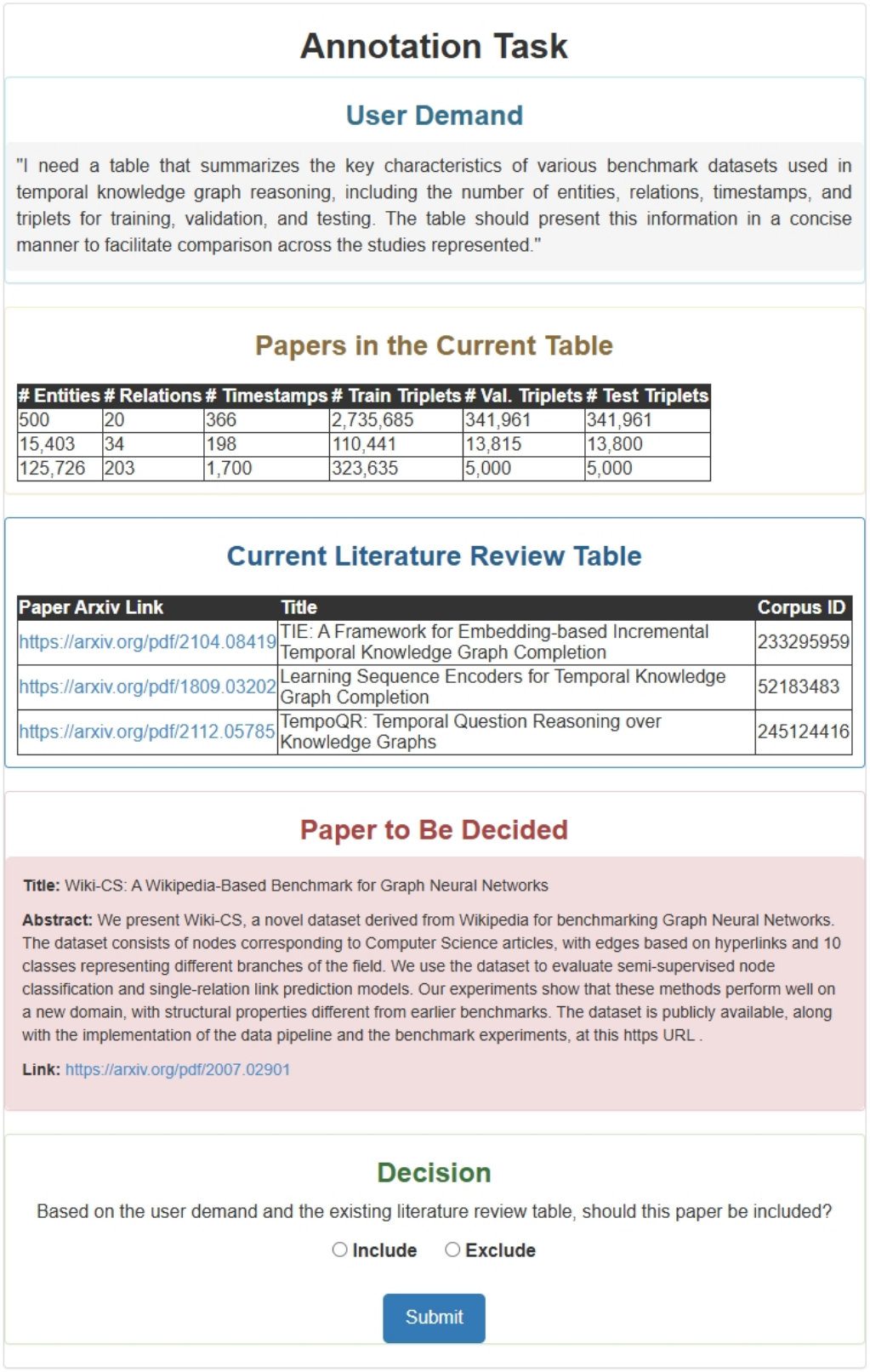

Figure 8: The annotation interface we used for collecting the gold labels for distractor papers.

Tasks	#Categories	Evaluation Metric
fine-grained	100	mean accuracy
face	9131	-

(a) Ground-truth table of the first pair of example.

Number of Images	Number of Subjects	Avg. Images per Subject	Number of Classes	Dataset Purpose
10,000	100	100	100	Fine-grained visual classification
3,310,000	9,131	362.6	9,131	Face recognition across variations

(b) Generated table of the first pair of example.

Problem	Description
Visual Reference Resolution	Capturing related visual region through an associative attention memory.
Visual Reference Resolution	Selectively referring dialogue history to refine the visual attention until referencing the answer.
Visual Reference Resolution	Establishing mapping of visual object and textual entities to exclude undesired visual content.
Visual-based Dialogue Strategies Optimization	Enhancing response generator with discriminator by RL reward.
Visual-based Dialogue Strategies Optimization	Maximizing the information gain while asking questions with a RL paradigm for explicit dialogue goals.
Pre-trained Vision Language Model-based VAD	Training unified Transformer encoder initialized by BERT with two visual training objectives.
Pre-trained Vision Language Model-based VAD	Utilizing GPT-2 to capture cross-modal semantic dependencies.
Unique Training Schemes-based VAD	Simulating Dual-coding theory of human cognition to adaptively find query-related information from the image.
Unique Training Schemes-based VAD	Asking questions to confirm the conjecture of models about the referent guided by human cognitive literature.

(c) Ground-truth table of the second pair of example.

ID	Method Used	Dataset	Problem Addressed	Performance Metric	Results Achieved	Model Type
5677543	Attention memory model	VisDial	Visual dialog with reference resolution	Answer prediction accuracy	16% improvement over state-of-the-art	Generative
54446647	Recursive Visual Attention mechanism	VisDial v0.9	Visual co-reference resolution	Mean Rank	State-of-the-art performance	Generative
236478107	Multimodal transformer with visual grounding	VisDial v0.9 and v1.0	Visual dialogue generation	BLEU	Achieves new state-of-the-art results	Generative
24537813	Adversarial learning with co-attention	VisDial	Visual dialog generation	Recall@5	+2.14% improvement over the previous best	Generative
196180698	Goal-oriented question generation model	GuessWhat?!	Goal-oriented visual dialogue	Accuracy	67.19% on GuessWhat?!	Generative
216562638	Vision-dialog transformer architecture	VisDial v0.9 and v1.0	Visual dialog	NDCG	New state-of-the-art performance	Generative and Discriminative
220045105	GPT-2 based architecture	AVSD	Video-grounded dialogue	BLEU	Outperforms existing approaches	Generative
208138178	Adaptive dual encoding framework	VisDial	Visual dialogue	Accuracy	State-of-the-art results	Generative
237491596	Beam search re-ranking algorithm	GuessWhat?!	Referential guessing games	Task accuracy	+4.35% improvement with re-ranking	Generative

(d) Generated table of the second pair of example.

Table 18: Case studies on the generation of literature review tables in ARXIV2TABLE.