HopWeaver: Synthesizing Authentic Multi-Hop Questions Across Text Corpora

Anonymous ACL submission

Abstract

Multi-Hop Question Answering (MHQA) is crucial for evaluating the model's capability to integrate information from diverse sources. However, creating extensive and high-quality MHQA datasets is challenging: (i) manual annotation is expensive, and (ii) current synthesis methods often produce simplistic questions or require extensive manual guidance. This paper introduces HopWeaver, the first automatic framework synthesizing authentic multi-hop questions from unstructured text corpora without human intervention. HopWeaver synthesizes two types of multi-hop questions (bridge and comparison) using an innovative approach that identifies complementary documents across corpora. Its coherent pipeline constructs authentic reasoning paths that integrate information across multiple documents, ensuring synthesized questions necessitate authentic multi-hop reasoning. We further present a comprehensive system for evaluating synthesized multi-hop questions. Empirical evaluations demonstrate that the synthesized questions achieve comparable or superior quality to human-annotated datasets at a lower cost. Our approach is valuable for developing MHQA datasets in specialized domains with scarce annotated resources.

1 Introduction

Integrating information from different sources shows the intelligence of Large Language Models (LLMs) and Retrieval-Augmented Generation (RAG) systems(Huang and Huang, 2024; Hu et al., 2024). Multi-Hop Question Answering (MHQA), as a critical benchmark for this ability, requires models to integrate information distributed across documents (Guo et al., 2024; Mavi et al., 2024). MHQA requires a model to connect intermediate entities or concepts across documents to infer answers. However, constructing extensive and high-quality MHQA datasets remains costly because manual annotation (Yang et al., 2018; Ho et al., 2020; Trivedi et al., 2022) struggles to cover diverse reasoning paths at scale and often introduces annotation bias (Klie et al., 2024; Wich et al., 2021).

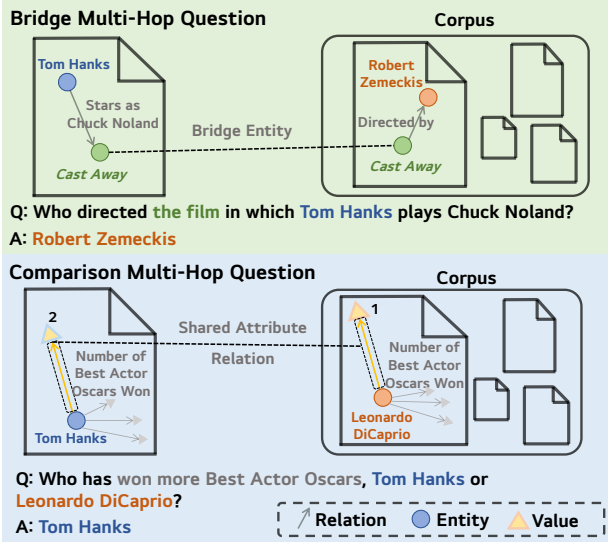

Figure 1: Examples of two multi-hop questions synthesized by HopWeaver: Bridge (top) and Comparison (bottom) question. These involve cross-document reasoning via a bridge entity or a shared attribute.

Recent studies have established synthesized data generation as a new paradigm for model training, testing, and evaluation (Lu et al., 2023; Guo and Chen, 2024). However, automatically synthesizing authentic questions that integrate multi-document information is particularly challenging for MHQA tasks (Mavi et al., 2022; Guo et al., 2024). Existing approaches require substantial manual intervention, including human-provided source documents (Fei et al., 2022; Xia et al., 2023), predefined question templates (He et al., 2024), or extraction of relationships from knowledge graphs and other structured data (Vuth et al., 2024; Chen et al., 2024). Some studies attempt to synthesize multi-hop-like questions but lack rigorous validation (Lupidi et al., 2024; Luo et al., 2024; Wu et al., 2024), result-

ing in "pseudo multi-hop" questions that can be resolved through single-document reasoning, and face two fundamental limitations: (i) to identify an entity that bridges distinct to complementary contexts, and (ii) to retrieve a complementary document rather than redundant factual repetitions. These limitations lead to low success rates and require manual post-filtering. Besides, existing evaluation approaches (Min et al., 2019; Ho et al., 2020; Trivedi et al., 2022) fail to evaluate synthesized MHQA datasets effectively.

We introduce **HopWeaver**, the first automatic framework that synthesizes multi-hop questions directly from raw corpora without any manual intervention. Building on established MHQA research (Yang et al., 2018; Ho et al., 2020; Trivedi et al., 2022; Yang et al., 2018; Ho et al., 2020), HopWeaver focuses on synthesizing two predominant question types: **bridge questions** (connecting facts across documents through intermediate entities) and **comparison questions** (contrasting attributes between entities). It employs an innovative retrieval mechanism that identifies authentically complementary documents and constructs reasoning paths that necessitate cross-document information integration (as shown in Figure 1).

We further develop a comprehensive evaluation system to evaluate synthesized questions by (i) LLM-as-judge, (ii) answerability and difficulty, and (iii) evidence-accessibility. In summary, HopWeaver enables the cost-effective synthesis of high-quality MHQA data, making it especially valuable in specialized domains where it can generate multi-hop questions directly from raw corpora without relying on human intervention or structured knowledge bases. (We will release the code for HopWeaver to facilitate further research in this area.) This work makes the following key contributions:

1. We propose HopWeaver, the first framework to automatically synthesize authentic multi-hop questions directly from raw corpora without human intervention or structured data.

2. We present a comprehensive data quality evaluation system for synthesized MHQA datasets with three complementary dimensions: LLM-as-judge, answerability/difficulty, and evidence-accessibility.

3. Empirical results demonstrating that: (i) our LLM-as-judge evaluation shows synthesized questions meet or exceed human-annotated benchmarks, (ii) answerability and difficulty evaluation confirms they require complex multi-hop reasoning, and (iii) evidence-accessibility evaluation proves they are well-grounded in source corpora.

2 Preliminaries

While MHQA exhibits various patterns (e.g., bridge, comparison, intersection, commonsense), analysis of major benchmarks (Yang et al., 2018; Ho et al., 2020; Trivedi et al., 2022) indicates that two types are fundamental and challenging: **bridge questions** and **comparison questions**. Building on existing research in MHQA, we formalize the key concepts used throughout this paper as follows:

2.1 Core Notation

Let $E = \{e_1, e_2, ..., e_n\}$ be the *entity set*, $R = \{r_1, r_2, ..., r_m\}$ the *relation set*, and $D = \{d_1, d_2, ..., d_l\}$ the *document set*. For each document $d_j \in D$, we define the function:

$$Trips(d_j) = \{t = (e_a, r_h, e_c) | t \text{ in } d_j\} \quad (1)$$

where each triplet $t = (e_a, r_h, e_c)$ represents a relational fact extracted from document d_j.

2.2 Bridge Question

Bridge questions link facts across documents through intermediate entities, necessitating cross-document reasoning. It aims to find a single sequential path P connecting a start entity e_s to a target entity e_t.

P is a sequence of k triplets (e_i, r_i, e_{i+1}) that form a path from e_1 to e_k:

$$P = \langle (e_1, r_1, e_2), ..., (e_{k-1}, r_{k-1}, e_k) \rangle \quad (2)$$

This framework imposes two constraints:

Fact Distribution Constraint: Each triplet in the path must come from exactly one document.

$$\forall (e_i, r_i, e_{i+1}) \in P, \exists! d_j \in D : (e_i, r_i, e_{i+1}) \in Trips(d_j) \quad (3)$$

No-Shortcut Constraint: No single document can bridge non-adjacent entities in the path.

$$\forall i, j : |i - j| > 1, \forall d \in D, \forall r \in R : (e_i, r, e_j) \notin Trips(d) \quad (4)$$

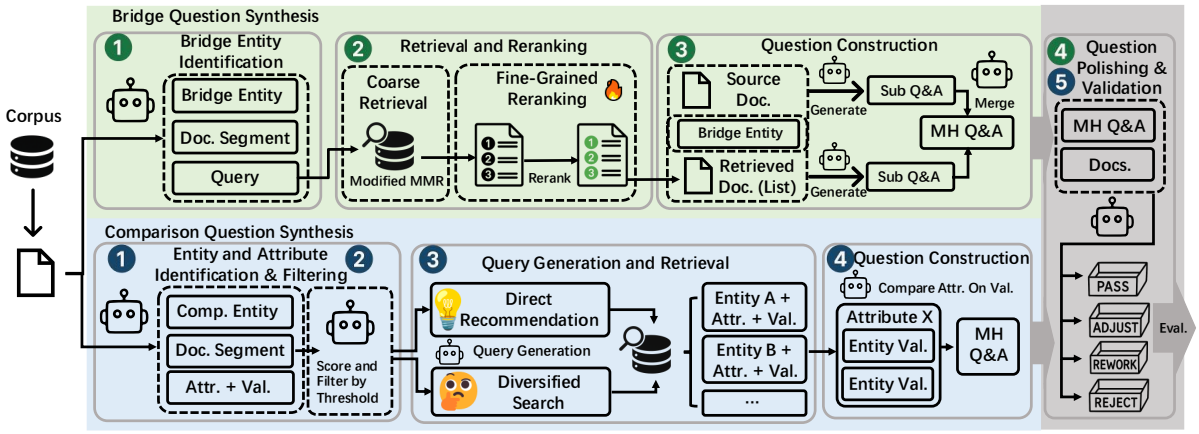

Figure 2: HopWeaver: Question Synthesis Framework

2.3 Comparison Question

Comparison questions contrast two entities (similar entities of the same category) by identifying their values for a specific relation, shared attribute, denoted as $r_c \in R$. This involves establishing attribute triplets for each entity:

$$t_1 = (e_1, r_c, v_1) \quad \text{and} \quad t_2 = (e_2, r_c, v_2) \tag{5}$$

Each triplet can be represented by a logical reasoning path:

$$P = \langle (e_1, r_1, e_1), \ldots, (e_{n-1}, r_n, e_v) \rangle \tag{6}$$

The paths may differ but must lead to attribute heads associated with r_c for effective comparison.

Distributed Source Constraint: A comparison requires multi-hop reasoning if no single document contains both facts t_1 and t_2. Let $D(t)$ be the set of documents stating triplet t. This means the sets of supporting documents must be disjoint:

$$D(t_1) \cap D(t_2) = \emptyset \tag{7}$$

3 Methodology

We introduce a framework for synthesizing two types of multi-hop questions: bridge questions and comparison questions (as shown in Figure 2).

3.1 Bridge Question Synthesis

Step 1: Bridge Entity Identification. First, a source document d_s is randomly sampled from the corpus. An LLM then processes d_s. The primary goal is to identify a reasonable *bridge entity* (e_b) that links disparate information contexts. The LLM achieves this goal by selecting a text segment (s_p) from d_s that provides concentrated textual context and then identifying e_b within s_p. Finally, based on e_b and s_p, the LLM formulates an optimized query (q), enabling the targeted retrieval of complementary documents in the next phase.

Step 2: Two-Stage Coarse-to-Fine Retrieval. This step takes the query q and source document d_s (from Step 1) as input. Its objective is to output a ranked list of k complementary documents, $D_t = \{d_t^1, \ldots, d_t^k\}$, which are identified through a two-stage retrieval strategy:

First, the **Coarse Retrieval** stage generates an initial candidate list. An initial set of documents is retrieved using q. Up to k candidates are greedily selected from this set using a modified Maximum Marginal Relevance (MMR) approach (Carbonell and Goldstein, 1998). As defined in Equation 8, this method balances query relevance ($\text{sim}(q, d_i)$) with dissimilarity to the source document d_s (via $-\text{sim}(d_i, d_s)$) and diversity among already selected documents in set S (via $-\max_{d_j \in S} \text{sim}(d_i, d_j)$), thereby promoting the selection of diverse and complementary contexts. The parameters $\lambda_1, \lambda_2, \lambda_3$ control this trade-off (as shown in Appendix E).

$$\begin{aligned} \text{Score}(d_i) = \lambda_1 \text{sim}(q, d_i) - \lambda_2 \text{sim}(d_i, d_s) \\ - \lambda_3 \max_{d_j \in S} \text{sim}(d_i, d_j) \end{aligned} \tag{8}$$

Next, the **Fine-grained Reranking** stage refines this candidate list. Candidate documents d_i are paired with q and re-scored by a fine-tuned reranker, yielding the final set D_t composed of the top k documents according to these new scores. The details of the reranker model and its fine-tuning methodology are provided in Section 3.3.

Step 3: Multi-Hop Question Construction. This step constructs a verifiable multi-hop ques-

tion by integrating information from the source document d_s and a selected complementary document d_t (from D_t identified in Step 2), using the bridge entity e_b as the pivot. The process consists of the following steps:

(a) **Sub-Question Generation**: To construct the final multi-hop question that requires grounding in both documents, two sequential sub-questions are generated: (i) Sub-Question 1 formulated from d_s with the bridge entity e_b as its answer; (ii) Sub-Question 2 generated from d_t with e_b in its question text, targeting information unique to d_t.

(b) **Multi-Hop Question Synthesis**: The two sub-questions are fused into a single, coherent multi-hop question. This final question is crafted to implicitly guide reasoning from d_s through e_b to d_t, without explicitly revealing e_b while necessitating multi-hop reasoning.

(c) **Validation and Iteration**: The synthesized question undergoes a validation process. If a valid multi-hop question is not successfully formed (e.g., due to flawed fusion, an invalid bridge connection, or entity ambiguity), or if the resulting question violates the Fact Distribution or No-Shortcut constraints (defined in Section 2.2), the current complementary document d_t is rejected. The question construction process then attempts to use the next ranked document from the list D_t.

The module generates a QA pair, a reasoning path ($d_s \rightarrow e_b \rightarrow d_t$), and sub-questions to ensure the interpretability and verifiability of each question. See Appendix B for an example of a synthesized bridge question and its generation details.

Step 4: Question Polishing and Validation. With the QA pair and its supporting segments generated in Step 3, we further enhance their quality by processing them through the Question Polishing and Validation module, detailed in Section 3.4.

3.2 Comparison Question Synthesis

Step 1: Entity and Attribute Identification. For each randomly sampled document d_a, the module:

- Identifies the primary subject entity e_a and its type (e.g., person, location, organization).
- Extracts 3-5 concise, factual attribute-value pairs (a, v_a) suitable for comparison (e.g., numeric, date, category).

Step 2: Filtering. To ensure concreteness and comparability, each entity and attribute is scored on a 1-5 scale (see Appendix D for detailed criteria), with only those meeting the threshold included in further steps.

Step 3: Query Generation and Retrieval. Based on its understanding of the source entity e_a and the filtered attributes from Step 2, the LLM generates queries and retrieves documents. The LLM selects Direct Recommendation when it possesses sufficient knowledge to identify a comparable entity, otherwise resorting to Diversified Search when its understanding is incomplete.

- **Direct Recommendation**: The LLM selects a representative attribute of e_a, recommends a comparable entity e_b, and generates a verification query to retrieve documents containing e_b with the same attribute.
- **Diversified Search**: The LLM generates three diverse retrieval queries to find other entities of the same type as e_a. These queries are executed, and their top-k results are merged to discover documents containing comparable entities.

This step outputs a list of retrieved documents.

Step 4: Question Construction. The system first identifies entity e_b within the retrieved document(s) and searches for a comparable attribute pair (a, v_a, v_b) where both entities have specific, factual values for the attribute a. The approach to finding this pair follows the strategy from Step 3:

- **Guided Comparison**: Following a Direct Recommendation, where a specific entity e_b and attribute a are specified, the system focuses on retrieving this exact pair.
- **Open Discovery**: Following a Diversified Search, the system iterates through the attributes of entity e_a to find the first valid comparable pair with any attribute of a discovered entity e_b.

When finding a comparable pair, the module generates a comparison QA pairs (e.g., *"Which has the higher a: e_a or e_b?" "e_a"*), both documents segments containing information of v_a and v_b, and corresponding reasoning path. An illustrative example of a comparison question generated by this process is provided in Appendix B.

Step 5: Question Polishing and Validation. The generated QA pair and its supporting segments are processed by the Question Polishing and Validation module (in Section 3.4) to ensure quality.

3.3 Fine-Tuning Reranker via Simulated Feedback

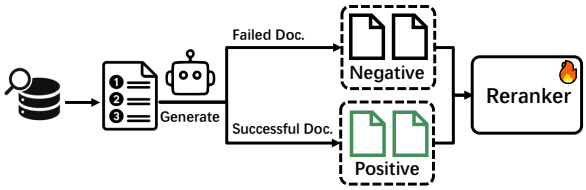

Figure 3: Fine-Tuning Reranker via Simulated Feedback

To enhance the fine-grained reranking stage (in Section 3.1, Step 2), we fine-tune the reranker using contrastive triples generated through simulating key steps of the bridge question synthesis process.

Generating Supervision Signals through Simulation. Our fine-tuning process begins with creating a labeled dataset directly from the bridge question synthesis process. We simulate this synthesis by retrieving a list of documents $\{d_i\}$ using our coarse retrieval methods. Each candidate d_i attempts to generate the connecting sub-questions required to link it with d_s through e_b. The success or failure of multi-hop question generation provides a supervision signal, with successful attempts yielding `positive example` (d^+) and failed attempts producing `negative example` (d^-).

Contrastive Learning Fine-Tuning of the Reranker. These positive and negative examples form the dataset for reranker fine-tuning. We construct contrastive training triples, typically (query $= e_b, d^+, d^-$), and train the reranker to distinguish complementary documents. This optimization is guided by a cross-entropy loss:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{\exp(f(e_b, d_i^+))}{\sum_{j=1}^{k}\exp(f(e_b, d_{ij}))}\right) \quad (9)$$

$f(e_b, d)$ is the score produced by the reranker for a query-document pair, N is the batch size, d_i^+ is the positive document in the i-th group, and d_{ij} represents all documents (one positive and $k-1$ negatives) in the j-th position within the i-th group for calculating the sum in the denominator.

This supervision signal, derived directly from the downstream task's success, assists the reranker to learn what constitutes a truly complementary document.

3.4 Question Polishing and Validation

The polisher module assesses and refines each multi-hop question (both Bridge and Comparison types) via structured prompts, generating one of four outcomes:

(i) **PASS**: accept the question.
(ii) **ADJUST**: apply minor wording or fluency improvements; output revised question and reasoning path.
(iii) **REWORKED**: perform substantial restructuring; output new question, reasoning path, and answer.
(iv) **REJECTED**: discard questions with irreparable flaws.

This step guarantees that each question (i) involves cross-document reasoning, (ii) hides the bridge entity (for Bridge questions), and (iii) maintains fluency without exposing intermediate steps.

4 Data Quality Evaluation System

Evaluating the quality of multi-hop questions requires a comprehensive evaluation system that extends beyond traditional metrics. We introduce a three-dimensional evaluation system designed to capture the critical attributes of high-quality MHQA datasets:

4.1 LLM-as-Judge Evaluation

Our evaluation employs an LLM-as-judge approach with a Likert scale (Liu et al., 2023) to evaluate each synthesized MHQA pair. This methodology incorporates recent advancements in question generation evaluation metrics (Fu et al., 2024b) through specific adaptations, thereby establishing a novel scoring framework tailored for multi-hop questions (in Appendix C).

While many studies compare LLM judges to human ratings (Ye et al., 2024), human evaluations are often inconsistent and biased (Chiang and Lee, 2023). Therefore, relying solely on human-LLM agreement might lead the LLM judge to inherit these limitations (Lee et al., 2025; Zheng et al., 2023). To avoid this, we evaluate LLM judges based on self-consistency, ensuring they produce stable, reproducible answers when evaluating the same input repeatedly (Lee et al., 2025). See Appendix C.2 for the rationale behind prioritizing LLM self-consistency.

To identify suitable LLM judges, we evaluated multiple models based on output stability. Results show proprietary LLMs like `GPT-4o` demonstrate strong performance across metrics, while open-source models such as `Gemma-3-27b` offer stable, cost-effective evaluation with better reproducibility. See Appendix C.3 for complete metrics, model specifications, selection rationale, and results with visualizations. We adopted the average score across selected judges as our final evaluation standard.

4.2 Answerability and Difficulty Evaluation

To evaluate the **answerability** and **difficulty** of our synthesized questions and their reliance on contextual evidence, we use multiple LLM solvers under two distinct conditions:

- **Q-Only**: The solver sees only the question. This setting primarily gauges the baseline answerability using the solver's internal knowledge and reasoning capabilities.
- **Q+Docs**: The solver receives all supporting documents for the question, simulating a golden retrieval scenario. This setting evaluates the question's answerability when all necessary evidence is available.

The performance improvement from the Q-Only to the Q+Docs indicates that (i) the question is challenging and requires contextual evidence rather than just pre-existing knowledge or superficial cues, (ii) the LLM-annotated golden evidence effectively supports a correct answer, confirming the question is answerable. These features are key signs of well-constructed multi-hop questions that test evidence integration.

4.3 Evidence-Accessiblity Evaluation

We examine whether annotated evidence for our synthesized question evidence is **accessible** in the corpus and evaluate the difficulty of its complete retrieval. Distinct retrieval methods are employed to fetch the top–k documents for each question and record retrieval metrics: (i) MAP (mean average precision), (ii) RECALL@K (proportion of golden evidence retrieved in top-k), (iii) NDCG@K (normalized discounted cumulative gain at k), and (iv) SUPPORT F1 (overlap between retrieved and golden evidence).

This comprehensive evaluation approach uses the recorded metrics to achieve two primary objectives: (i) to gauge the accessibility of individual evidence documents (using RECALL@K, e.g., @20), which is crucial for verifying corpus grounding and evaluating retrieval ranking quality (via MAP and NDCG@K); and (ii) to identify specific difficulties in multi-source evidence assembly, indicated by SUPPORT F1 (complete-set retrieval accuracy) relative to individual document recall (RECALL@K). This detailed analysis provides a vital retrieval baseline for our dataset, enabling a clearer interpretation of MHQA performance by clearly distinguishing retrieval challenges from reasoning demands.

5 Experiments

HopWeaver is evaluated using the English Wikipedia corpus and four LLM generators with different scales and performances for synthesis. We compare the synthesized question with three human-annotated MHQA datasets. See Appendix E for the complete experimental setup (including corpus details, model specifications, and evaluation metrics) and Appendix A for cost analysis.

5.1 Main Quality Evaluation

Table 1: Quality evaluation of multi-hop questions (on 100 samples, five LLM judges). Multi-Hop (%) shows the proportion of questions authentically involving information from multiple documents. Avg. Score represents quality on a 1-5 scale (1=Very Poor, 5=Very Good) across multiple evaluation criteria (in Appendix C).

Generation Source	Bridge Questions		Comparison Questions	
	Multi-Hop (%)	Avg. Score	Multi-Hop (%)	Avg. Score
HopWeaver (Ours)				
w/ Gemini-2.5-flash	96.4	**4.27**	**98.6**	**4.45**
w/ QwQ-32B	**98.9**	4.23	97.4	4.40
w/ Qwen-14B	96.9	4.09	95.9	4.36
w/ GLM-4-9B-0414	89.8	3.87	93.9	4.26
Human Datasets (Baselines)				
HotpotQA	92.8	4.23	95.6	4.20
2WikiMultiHopQA	92.8	4.04	97.6	4.42
MuSiQue	91.2	3.78	N/A	N/A

Our quality evaluation results in Table 1 contain the proportion of authentic multi-hop questions and their average scores. When employing the proprietary LLM `Gemini-2.5-flash`, HopWeaver achieves exceptional performance (98.6% multi-hop rate and 4.47 average score for Comparison questions; 96.4% and 4.27 for Bridge questions), surpassing all evaluated human-annotated datasets. This demonstrates HopWeaver's capacity to produce data that advance MHQA research.

To evaluate HopWeaver's performance with more accessible and reproducible setups, we also tested three leading open-source LLMs of varying scales (`QwQ-32B`, `Qwen-14B`, `GLM-4-9B-0414`). These models also enable HopWeaver to synthesize

high-quality questions that rival or exceed human datasets. For instance, `QwQ-32B` achieves a 98.9% multi-hop rate and a 4.23 average score for Bridge questions. Even the smaller `GLM-4-9B-0414` yields a high percentage of valid multi-hop questions (89.8% for Bridge, 93.9% for Comparison), offering a cost-effective solution for large-scale synthesis. This consistent performance across diverse models underscores HopWeaver's effectiveness. While powerful closed-source models establish the upper bound of achievable quality, the results with open-source alternatives make them excellent choices for researchers prioritizing cost-efficiency and scientific reproducibility, mitigating concerns about the instability or potential deprecation of proprietary LLMs.

A multi-dimensional analysis is conducted to compare the average scores across key quality dimensions for HopWeaver-synthesized questions (using different LLMs) against human datasets. Figure 4 shows that the question synthesized by Hopweaver surpasses the benchmark of human datasets in most dimensions, especially in **logical sophistication** and **information integration**, although the top human dataset holds a marginal advantage in conciseness.

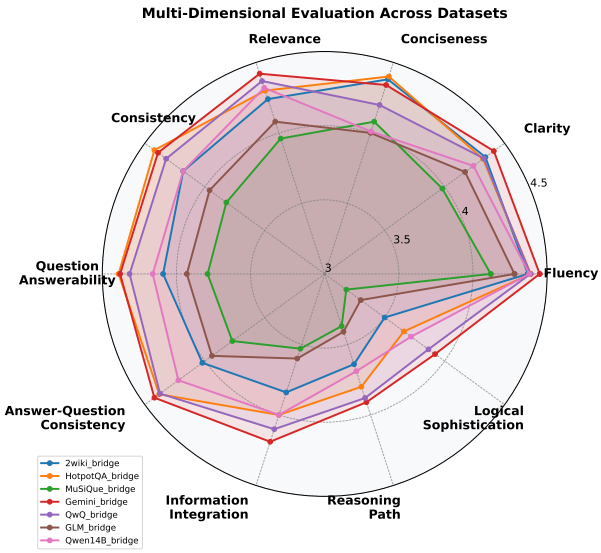

Figure 4: Multi-dimensional quality evaluation to compare HopWeaver-synthesized questions (by different LLMs) against baseline datasets across key criteria, and scale truncated to [3.0, 4.5] for visualization clarity (full scale: [1, 5]).

5.2 Answerability and Difficulty Evaluation

We evaluate various LLMs in both "Question-Only" (Q-Only) and "Question + Golden Documents" (Q+Docs) settings, as described in Section 4.2. The results, presented in Table 2, demonstrate a significant performance improvement across all models when the golden supporting documents are provided. For instance, GPT-4o's Exact Match (EM) score rose from 29.0% to 51.0% on Bridge questions (Q-Only to Q+Docs) and jumped from 69.0% to 94.0% on Compare questions (Q-Only to Q+Docs).

This substantial gain confirms two critical aspects aligned with our evaluation goals: (i) the questions involve multi-hop reasoning and are difficult to answer based solely on the models' internal knowledge. (ii) The golden documents synthesized by HopWeaver are effective and contain the necessary information for models to deduce the correct answers, confirming the questions' answerability given appropriate evidence. Besides, we manually inspected GPT-4o's errors (Q+Docs) to further validate question quality. This inspection reveals that errors stemmed from model inference issues or ambiguities, not flaws in the synthesized questions, confirming their answerability and correctness.

Most notably, smaller models like Qwen3-8B show substantial performance gains, with its F1 score increasing dramatically from 22.3% to 60.2% on bridge questions when gold documents are included. This clearly indicates that models with less extensive internal knowledge can still perform robust reasoning when relevant contextual evidence is properly provided by HopWeaver.

Furthermore, the consistent ranking of model performance suggests that our dataset serves as a reliable benchmark for evaluating and comparing the reasoning capabilities of different LLMs in multi-hop scenarios.

Table 2: QA Results, comparing model performance with Question Only (Q-Only) and Question + Golden Docs (Q+Docs), 100 samples.

Model	Bridge				Compare			
	Q-Only (%)		Q+Docs (%)		Q-Only (%)		Q+Docs (%)	
	EM	F1	EM	F1	EM	F1	EM	F1
GPT-4o	**29.0**	**40.5**	**51.0**	**65.0**	**69.0**	**70.5**	**94.0**	**94.2**
Claude-3.7-Sonnet	23.0	33.0	50.0	62.0	47.0	50.2	91.0	93.4
Llama-3.3-70B	18.0	30.6	45.0	60.9	49.0	51.8	74.0	78.9
Qwen3-8B	10.0	22.3	47.0	60.2	50.0	55.4	62.0	66.0

5.3 Evidence-Accessiblity Evaluation

We evaluated the synthesized dataset (by Gemini-2.5) using BM25, GTE, and E5 retrievers. Our evaluation requires retrievers to identify the exact document sources used to synthesize each question.

The evidence-accessibility evaluation (Table 3)

highlights several key findings regarding the dataset: (i) Evidence documents are largely accessible by retriever within the corpus, as demonstrated by Recall@k (e.g., BM25 Recall@20 of 0.7050), affirming the questions' strong corpus-grounding; (ii) retrieving the complete set of necessary evidence documents simultaneously remains challenging (e.g., low Support F1 scores: 0.17-0.22), despite individual document accessibility. This disparity underscores the genuine multi-hop nature of the questions, which necessitate integrating information from multiple, distinct sources; (iii) the dataset effectively discriminates between retrieval strategies (e.g., BM25 > GTE > E5), demonstrating its utility as a benchmark for evaluating multi-hop retrieval systems.

Table 3: Evidence-Accessiblity Evaluation Results. Evaluating different retrievers on the HopWeaver synthesized dataset.

Method	MAP	Recall@5	Recall@10	Recall@20	NDCG@5	NDCG@10	Support F1
BM25	**0.5605**	**0.6150**	**0.6650**	**0.7050**	**0.6305**	**0.6510**	**0.2217**
GTE	0.5092	0.5600	0.5900	0.6250	0.5828	0.5946	0.1967
E5	0.4107	0.4600	0.5150	0.5800	0.4720	0.4943	0.1717

5.4 Ablation Study

5.4.1 Reranker Efficiency

To evaluate our fine-tuned reranker's efficiency (in Section 3.3), we compare retrieval strategies for Bridge question synthesis using two metrics: *Success Rate* (percentage of documents yielding valid questions) and *Average Attempts for Success* (attempts needed to find successful pairs).

Table 4 reveals progressively stronger results from standard dense retrieval ('standard') and MMR diversity ('diverse') to zero-shot reranking ('diverse + rerank (ZS)'). The optimal performance comes from our fine-tuned reranker ('diverse + rerank (FT)'), delivering both superior success rates and requiring fewer document retrieval attempts. This demonstrates that fine-tuning substantially improves question synthesis efficiency and reduces computational costs.

Table 4: Ablation study on the effectiveness of the fine-tuned reranker for Bridge question synthesis.

Retrieval Strategy	Success Rate (%) ↑	Avg. Attempts ↓
Standard Dense Retrieval	70.1	1.59
Diverse (MMR)	70.9	1.62
Diverse + Rerank (ZS)	74.0	1.32
Diverse + Rerank (FT - Ours)	**75.3**	**1.17**

5.4.2 Polisher Module Effectiveness

We investigate the contribution of the Polisher module (Section 3.4) to the final question quality. We use LLM-as-judge to compare the quality assessment of questions between directly synthesized by LLMs ('Original') and the questions after being processed by the Polishing module ('Polished').

Table 5 shows that the Polisher module improves both the multi-hop validity rate and the average quality score for both Bridge and Comparison questions. This demonstrates the importance of the refinement and validation step in ensuring high-quality synthesized data. Notably, our analysis reveals a differentiated impact based on the capabilities of the generator LLMs. For stronger models like Gemini-2.5-flash, the Polisher's improvements are modest, while smaller models such as GLM-4-9B exhibit more substantial gains (e.g., bridge question score rising from 3.71 to 3.87)

This highlights the Polisher module's value in a resource-optimized pipeline, enabling smaller generator models to achieve high-quality outputs through targeted refinement rather than scaling up model parameters.

Table 5: Ablation study on the effectiveness of the Polisher module across different generator LLMs.

Generator	Version	Bridge Questions		Comparison Questions	
		Multi-Hop (%)	Avg. Score	Multi-Hop (%)	Avg. Score
Gemini	Original	95.2	4.26	98.0	4.42
	Polished	96.4	4.27	98.6	4.45
QwQ-32B	Original	97.6	4.20	97.2	4.36
	Polished	98.9	4.23	97.4	4.40
Qwen3-14B	Original	96.8	4.03	94.0	4.34
	Polished	96.9	4.09	95.9	4.36
GLM-4-9B	Original	84.5	3.71	92.8	4.13
	Polished	89.8	3.87	93.9	4.25

6 Conclusion

We presented HopWeaver, a fully automatic framework for synthesizing authentic multi-hop questions (bridge and comparison) from raw text corpora. Our experiments demonstrate that HopWeaver meets or exceeds human-level benchmarks across multiple evaluation dimensions and scales effectively with various LLMs. These capabilities make HopWeaver a practical solution for constructing complex MHQA datasets in domains where human annotation is limited. Our comprehensive evaluation system also provides valuable metrics for evaluating synthesized question quality, reducing cost compared to manual annotation.

Limitations

While HopWeaver is capable of synthesizing authentic multi-hop questions, it currently focuses on the two-hop bridge and comparison reasoning patterns. Future work could extend the framework to generate a broader range of more complex questions involving longer reasoning chains or mixed reasoning types.

References

Anna Bavaresco, Raffaella Bernardi, Leonardo Bertolazzi, Desmond Elliott, Raquel Fernández, Albert Gatt, Esam Ghaleb, Mario Giulianelli, Michael Hanna, Alexander Koller, André F. T. Martins, Philipp Mondorf, Vera Neplenbroek, Sandro Pezzelle, Barbara Plank, David Schlangen, Alessandro Suglia, Aditya K. Surikuchi, Ece Takmaz, and Alberto Testoni. 2024. Llms instead of human judges? A large scale empirical study across 20 NLP evaluation tasks. *CoRR*, abs/2406.18403.

Jaime G. Carbonell and Jade Goldstein. 1998. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *SIGIR '98: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 24-28 1998, Melbourne, Australia*, pages 335–336. ACM.

Ruirui Chen, Weifeng Jiang, Chengwei Qin, Ishaan Singh Rawal, Cheston Tan, Dongkyu Choi, Bo Xiong, and Bo Ai. 2024. LLM-based multi-hop question answering with knowledge graph integration in evolving environments. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 14438–14451, Miami, Florida, USA. Association for Computational Linguistics.

David Cheng-Han Chiang and Hung-yi Lee. 2023. Can large language models be an alternative to human evaluations? 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 15607–15631. Association for Computational Linguistics.

Zichu Fei, Qi Zhang, Tao Gui, Di Liang, Sirui Wang, Wei Wu, and Xuanjing Huang. 2022. CQG: A simple and effective controlled generation framework for multi-hop question generation. 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 6896–6906. Association for Computational Linguistics.

Emilio Ferrara. 2023. Should chatgpt be biased? challenges and risks of bias in large language models. *First Monday*, 28(11).

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

Jinlan Fu, See-Kiong Ng, Zhengbao Jiang, and Pengfei Liu. 2024a. Gptscore: Evaluate as you desire. 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 6556–6576. Association for Computational Linguistics.

Weiping Fu, Bifan Wei, Jianxiang Hu, Zhongmin Cai, and Jun Liu. 2024b. QGEval: Benchmarking multi-dimensional evaluation for question generation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 11783–11803, Miami, Florida, USA. Association for Computational Linguistics.

Shasha Guo, Lizi Liao, Cuiping Li, and Tat-Seng Chua. 2024. A survey on neural question generation: Methods, applications, and prospects. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI 2024, Jeju, South Korea, August 3-9, 2024*, pages 8038–8047. ijcai.org.

Xu Guo and Yiqiang Chen. 2024. Generative AI for synthetic data generation: Methods, challenges and the future. *CoRR*, abs/2403.04190.

Jie He, Nan Hu, Wanqiu Long, Jiaoyan Chen, and Jeff Z. Pan. 2024. MINTQA: A multi-hop question answering benchmark for evaluating llms on new and tail knowledge. *CoRR*, abs/2412.17032.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Linmei Hu, Zeyi Liu, Ziwang Zhao, Lei Hou, Liqiang Nie, and Juanzi Li. 2024. A survey of knowledge enhanced pre-trained language models. *IEEE Trans. Knowl. Data Eng.*, 36(4):1413–1430.

Yizheng Huang and Jimmy Huang. 2024. A survey on retrieval-augmented text generation for large language models. *CoRR*, abs/2404.10981.

Jiajie Jin, Yutao Zhu, Xinyu Yang, Chenghao Zhang, and Zhicheng Dou. 2024. Flashrag: A modular toolkit for efficient retrieval-augmented generation research. *CoRR*, abs/2405.13576.

Jan-Christoph Klie, Richard Eckart de Castilho, and Iryna Gurevych. 2024. Analyzing dataset annotation quality management in the wild. *Comput. Linguistics*, 50(3):817–866.

Klaus Krippendorff. 2018. *Content analysis: An introduction to its methodology*. Sage publications.

Noah Lee, Jiwoo Hong, and James Thorne. 2025. Evaluating the consistency of LLM evaluators. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 10650–10659, Abu Dhabi, UAE. Association for Computational Linguistics.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: NLG evaluation using gpt-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 2511–2522. Association for Computational Linguistics.

Adian Liusie, Potsawee Manakul, and Mark J. F. Gales. 2024. LLM comparative assessment: Zero-shot NLG evaluation through pairwise comparisons using large language models. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2024 - Volume 1: Long Papers, St. Julian's, Malta, March 17-22, 2024*, pages 139–151. Association for Computational Linguistics.

Yingzhou Lu, Huazheng Wang, and Wenqi Wei. 2023. Machine learning for synthetic data generation: a review. *CoRR*, abs/2302.04062.

Haohao Luo, Yang Deng, Ying Shen, See-Kiong Ng, and Tat-Seng Chua. 2024. Chain-of-exemplar: Enhancing distractor generation for multimodal educational question generation. 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 7978–7993. Association for Computational Linguistics.

Alisia Lupidi, Carlos Gemmell, Nicola Cancedda, Jane Dwivedi-Yu, Jason Weston, Jakob Foerster, Roberta Raileanu, and Maria Lomeli. 2024. Source2synth: Synthetic data generation and curation grounded in real data sources. *CoRR*, abs/2409.08239.

Vaibhav Mavi, Anubhav Jangra, and Adam Jatowt. 2022. A survey on multi-hop question answering and generation. *CoRR*, abs/2204.09140.

Vaibhav Mavi, Anubhav Jangra, and Adam Jatowt. 2024. Multi-hop question answering. *Found. Trends Inf. Retr.*, 17(5):457–586.

Sewon Min, Eric Wallace, Sameer Singh, Matt Gardner, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019. Compositional questions do not necessitate multi-hop reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4249–4257, Florence, Italy. Association for Computational Linguistics.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. 9835 musique: Multi-hop questions via single-hop question composition. *Trans. Assoc. Comput. Linguistics*, 10:539–554.

Nakanyseth Vuth, Gilles Sérasset, and Didier Schwab. 2024. KGAST: From knowledge graphs to annotated synthetic texts. In *Proceedings of the 1st Workshop on Knowledge Graphs and Large Language Models (KaLLM 2024)*, pages 43–55, Bangkok, Thailand. Association for Computational Linguistics.

Maximilian Wich, Christian Widmer, Gerhard Hagerer, and Georg Groh. 2021. Investigating annotator bias in abusive language datasets. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021), Held Online, 1-3 September, 2021*, pages 1515–1525. INCOMA Ltd.

Ian Wu, Sravan Jayanthi, Vijay Viswanathan, Simon Rosenberg, Sina Pakazad, Tongshuang Wu, and Graham Neubig. 2024. Synthetic multimodal question generation. In *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, pages 12960–12993. Association for Computational Linguistics.

Zehua Xia, Qi Gou, Bowen Yu, Haiyang Yu, Fei Huang, Yongbin Li, and Cam-Tu Nguyen. 2023. Improving question generation with multi-level content planning. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 800–814. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2369–2380. Association for Computational Linguistics.

Jiayi Ye, Yanbo Wang, Yue Huang, Dongping Chen, Qihui Zhang, Nuno Moniz, Tian Gao, Werner Geyer, Chao Huang, Pin-Yu Chen, Nitesh V. Chawla, and Xiangliang Zhang. 2024. Justice or prejudice? quantifying biases in llm-as-a-judge. *CoRR*, abs/2410.02736.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

A Cost Analysis

The tests were conducted on gemini-2.5-flash-preview-04-17 (the most expensive model we have tested). The following table 6 summarizes the token consumption for question generation and LLM-as-Judge evaluation (on GPT-4o).

Table 6: Token Consumption Analysis

Process	Avg. Requests/Calls	Input Tokens (Per. Req.)	Output Tokens (Per. Req.)
Question Synthesis	7.6	1529.97	231.32
LLM-as-Judge Evaluation	5	1885.53	83.00

For closed-source models, API calls can be made via their respective platforms. For example, based on the consumption data in this table, synthesizing 1000 multi-hop questions (7600 times requests) using Gemini 2.5 Flash (assuming API pricing of \$0.15 per million input tokens and \$3.50 per million output tokens) would cost approximately \$7.90. Similarly, using GPT-4o (assuming API pricing of \$2.5 per million input tokens and \$10 per million output tokens) as a single evaluation model to evaluate 1000 synthesized questions would cost approximately \$27.72. For open-source models, we utilized a setup with 4x A100 (40GB VRAM) GPUs for deployment and inference with bf16 precision. For some open-source models, due to considerations of complex deployment engineering efforts, we used APIs via OpenRouter while still ensuring high reproducibility.

The result implies that the cost of synthesizing a dataset with thousands of entries is significantly lower than manual annotation; if small-scale open-source models are used, these generation costs can be almost negligible. Furthermore, if multiple LLMs are required for large-scale evaluation of dataset quality, the corresponding computational power or financial costs must also be taken into consideration.

B Examples of Synthesized Questions

B.1 Bridge Question Example

Figure 5 shows an example of a synthesized bridge question detailing the involved reasoning path and source information.

B.2 Comparison Question Example

An example of a comparison question is presented in Figure 6, highlighting the entities and attributes under comparison.

Source Document (**Anatomy Snippet**):
"...medial tendinous margins of the crura... meet ... to form an arch ... known as the **median arcuate ligament**..."
(Context: Defines the anatomical structure)

Retrieved Document (**Pathology Snippet**):
"**Median arcuate ligament syndrome (MALS)**... is a condition characterized by... compression ... by the **median arcuate ligament**."
(Context: Links the structure to the medical condition)

Bridge Entity: **median arcuate ligament**

Reasoning Path :
1. Identify Structure (from Source): The anatomical structure formed is the **median arcuate ligament**.
2. Link to Condition (from Target): Compression by the **median arcuate ligament** leads to **Median arcuate ligament syndrome**.

Question:
What medical condition is attributed to the compression caused by the anatomical structure formed by the meeting of the medial tendinous margins of the diaphragm's crura?

Answer:
Median arcuate ligament syndrome (MALS)

Figure 5: An example of a bridge question synthesized by HopWeaver. The figure illustrates the source documents, the identified bridge entity, the reasoning steps, and the final generated question-answer pair.

C Evaluation Criteria for LLM-as-Judge

C.1 Pointwise Scoring Framework

To assign an absolute quality score to each synthesized question-answer pair and ensure a rigorous, multi-faceted evaluation, we employ a *pointwise* scoring approach. This method allows for the independent evaluation of each item against a predefined set of criteria by a LLM judge (Liu et al., 2023; Fu et al., 2024a). We opted for pointwise scoring over pairwise comparison (Liusie et al., 2024) because our synthesized questions are non-parallel and vary significantly in difficulty, making it challenging to establish fair comparative benchmarks necessary for pairwise approaches. The detailed criteria for our pointwise evaluation are organized into three main categories:

- **Multi-Hop QA Rule Dimension**: This is a binary (Yes/No) evaluation determining if the question authentically involves reasoning across multiple documents, where information from one document is necessary to understand or utilize information in another, and the answer cannot be derived from any single document. This dimension is paramount; a "No" indicates a fundamental failure.
- **Linguistic Dimensions**: These evaluate the

📄 Source Document (Composer Biography Snippet):
"**Mihály Mosonyi** (**4 September 1815** in Boldogasszony, Austria-Hungary – 31 October 1870 in Budapest) was a Hungarian composer."
(Context: The biographical information of Hungarian composer Mihály Mosonyi)

📄 Retrieved Document (Composer Biography Snippet):
"The future composer, only son of **Adam Liszt** and his wife Maria Anna, was born here on 22 **October 1811**."
(Context: Details Liszt's birthplace and childhood)

🔗 Compared Attribute: **Date of Birth**

🔍 Reasoning Path:
1. Identify Information (from Source): Mihály Mosonyi was born on September 4, 1815
2. Identify Information (from Target): Franz Liszt was born on October 22, 1811
3. Compare Dates: 1811 is earlier than 1815

❓ Question:
Which composer has an earlier Date of Birth: **Mihály Mosonyi** or **Franz Liszt**?

💡 Answer:
Franz Liszt

Figure 6: An example of a comparison question synthesized by HopWeaver. This showcases the two entities being compared, the specific attribute, the source evidence snippets, and the resulting question.

quality of question presentation, ensuring understandability and precision. Criteria include:

- *Fluency*: Grammatical correctness and coherence.
- *Clarity*: Unambiguous and precise expression.
- *Conciseness*: Absence of redundant information.

- **Task-Oriented Dimensions**: These evaluate the functional and logical aspects of the question-answer pair within the provided document context. Criteria include:
  - *Relevance*: Appropriateness to the given passages and focus on key information.
  - *Consistency*: Strict adherence of the question's information to the source passages, free from contradictions or hallucinations, however subtle.
  - *Question Answerability*: Whether the question can be clearly and unambiguously answered solely from the provided passages.
  - *Answer-Question Consistency*: Accuracy and completeness of the answer in addressing the question.
  - *Information Integration Ability*: Successful and logical integration of information from multiple documents, without forcing unnatural connections.
  - *Reasoning Path Guidance*: Clear direction for a multi-step reasoning process.
  - *Logical Sophistication*: Clever design requiring multi-step thinking, free from logical gaps or fallacies, presenting a genuinely challenging and sound multi-hop problem.

  Particularly, dimensions such as *Consistency*, *Information Integration Ability*, and *Logical Sophistication* are critical. Flaws in these areas are heavily penalized, reflecting their significance in authentic multi-hop question quality.

For scoring the Linguistic and Task-oriented dimensions, a Likert-like scale (Very Poor, Poor, Fair, Good, Very Good) is employed. However, the LLM judge is instructed to adopt a **skeptical default stance** and interpret these scale points with heightened strictness, as summarized below, to minimize subjective bias and ensure only high-quality items receive favorable scores (Liu et al., 2023):

- **Very Poor (Unacceptable)**: Fundamentally flawed (e.g., not truly multi-hop, severe contradictions, unanswerable).

- **Poor (Weak/Barely Usable)**: Obvious, major flaws requiring significant revision (e.g., weak/forced logic, inconsistencies).

- **Fair (Acceptable/Passable)**: Basic requirements met but with notable flaws or room for improvement; signifies minimum adequacy only, not a positive endorsement.

- **Good**: Well-designed, logically clear, fluent, and meets multi-hop criteria without obvious flaws.

- **Very Good (Excellent/Outstanding)**: Exemplary design with deep logic, precision, and rigor.

This stringent evaluation mechanism, with a directive to assign lower ratings ('Poor' or 'Very Poor')

when significant flaws are present (especially logical ones), effectively filters out low-quality or trivially multi-hop questions. A question with significant logical flaws cannot achieve a 'Good' or 'Very Good' rating overall, even if linguistically sound.

C.2 Beyond Human Alignment: The Case for LLM Self-Consistency

Human judgments, while valuable, exhibit limitations that challenge their role as the sole benchmark for aligning large language models (LLMs). First, inter-rater reliability in subjective tasks is often low; for example, human evaluations of dialogue quality show poor agreement (e.g., Krippendorf's $\alpha = 0.33$ on PersonaChat, indicating fair agreement; $\alpha = 0.08$ on WMT 2020 Zh-En, indicating slight agreement; $\alpha = 0.49$ on QAGS, indicating moderate agreement, (Bavaresco et al., 2024)). Similarly, in MHQA tasks, which often span multiple domains and require complex reasoning, human judgments are likely to be inconsistent due to the subjective nature of criteria like question clarity, relevance, and logical sophistication. Second, human ratings are prone to systematic biases—such as fatigue, cultural preferences, or contextual misunderstandings—which introduce variability and undermine evaluation reliability. Indeed, research into human perceptions of LLM outputs reveals that factors unrelated to intrinsic quality, such as perceived LLM sentience or anthropomorphism, can influence human evaluations (Lee et al., 2025; Zheng et al., 2023; Chiang and Lee, 2023).

Furthermore, much of the current work on LLM-as-judge focuses on achieving high correlation with human preferences or ratings (Ye et al., 2024). While aligning with human intuition is a desirable goal, an overemphasis on mimicking human scores can inadvertently lead LLM judges to replicate the aforementioned human biases and inconsistencies. If human agreement itself is a noisy or unstable signal, then LLM judges optimized solely for human-LLM consistency may inherit these limitations rather than serving as a more objective or stable evaluation instrument. This is particularly problematic when the goal is to create a scalable and reliable evaluation framework(Lee et al., 2025; Zheng et al., 2023).

Given these constraints, we contend that while human feedback remains a crucial component in the broader LLM development lifecycle, pursuing perfect alignment between LLM judge scores and raw human judgments as the *primary* Evaluation metric for the judge itself is neither always feasible nor universally desirable for all evaluation tasks. Instead, we propose prioritizing the **self-consistency** of LLMs—defined as their ability to deliver stable, reproducible outputs for identical inputs under controlled conditions—as a foundational criterion for selecting qualified judge models. This shift towards emphasizing demonstrable reliability in the LLM judge's own behavior ensures a more standardized and robust evaluation framework, mitigating the risk of amplifying the inherent shortcomings of human-based assessments when seeking fine-grained, repeatable quality scores.

C.3 LLM Reliability: Metrics and Evaluation Results

We posit that a trustworthy judge must produce stable outputs under identical conditions. Given each item, we sample $N = 5$ independent runs at temperature $T = 0$ to ensure output stability, as lower temperatures are suitable for evaluation tasks.

Metrics To provide a comprehensive evaluation of LLM judge reliability from multiple perspectives, we employ three complementary metrics: Avg. Intra-item SD measures the direct stability of scores, while Krippendorff's Alpha and Fleiss' Kappa evaluate the statistical significance of inter-run agreement corrected for chance.

(i) Avg. Intra-item SD measures score volatility for each item across N repeated runs:

$$\mathrm{SD}_i = \mathrm{std}(\{s_i^{(1)}, \ldots, s_i^{(N)}\}) \tag{10}$$

$$\mathrm{AvgSD} = \frac{1}{M} \sum_{i=1}^{M} \mathrm{SD}_i \tag{11}$$

where $s_i^{(j)}$ is the score for item i in run j, and M is the total number of items.

(ii) Krippendorff's α (Krippendorff, 2018) treats the N runs as raters and measures agreement beyond chance for various data types:

$$\alpha = 1 - \frac{D_o}{D_e} \tag{12}$$

where D_o is the observed disagreement and D_e is the expected disagreement by chance.

(iii) Fleiss' Kappa (κ) (Fleiss, 1971) measures inter-rater agreement for categorical ratings, evaluating reliability among multiple raters (our N runs):

$$\kappa = \frac{\bar{P} - \bar{P}_e}{1 - \bar{P}_e} \tag{13}$$

where $\bar{P}$ is the mean proportion of observed agreement among raters, and $\bar{P}_e$ is the mean proportion of agreement expected by chance.

Based on these reliability metrics, we evaluate several open-source and proprietary LLMs on a held-out subset ($M = 100$). The model with higher agreement metrics (α, κ) and lower AvgSD values is selected as the LLM judge. To avoid self-enhancement bias (Ferrara, 2023), the chosen judge never scores its own generations or those from close variants.

Figure 7: Heatmap visualizing AvgSD scores across different LLMs and evaluation dimensions.

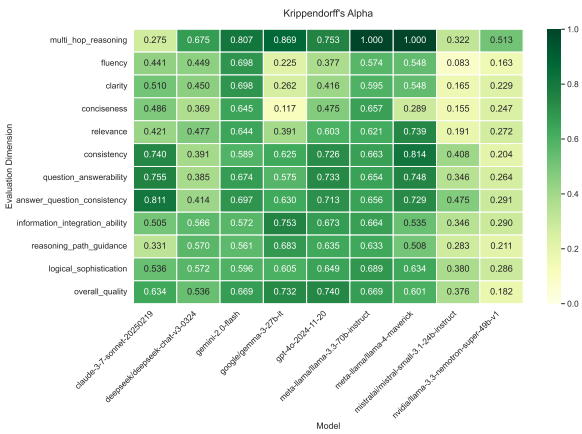

Figure 8: Heatmap visualizing Krippendorff's Alpha scores across different LLMs and evaluation dimensions.

Final LLM Judge Ensemble Considering a balance of evaluation performance (as indicated by the reliability metrics defined earlier in this appendix and further detailed in Table 7 presented below) and operational costs, we selected an ensemble of LLMs to serve as our final judges:

- `claude-3-7-sonnet-20250219`

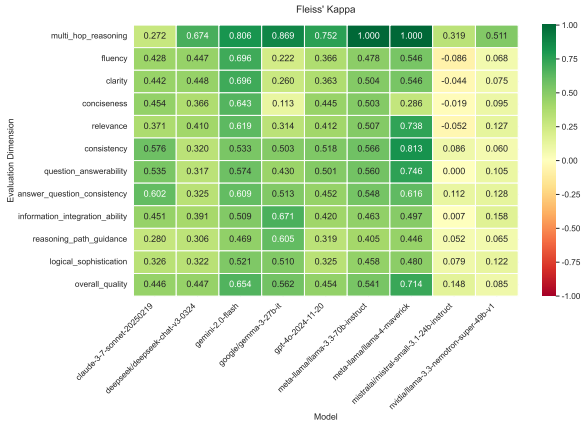

Figure 9: Heatmap visualizing Fleiss' Kappa scores across different LLMs and evaluation dimensions.

- `gpt-4o-2024-11-20`
- `gemini-2.0-flash`
- `google/gemma-3-27b-it`
- `meta-llama/llama-3.3-70b-instruct`

The use of this diverse set of models, including both proprietary and open-source options, aims to provide a robust and comprehensive evaluation, while also considering the reproducibility and accessibility of the evaluation process. The average scores from this ensemble are used for the final quality evaluation of the synthesized multi-hop questions.

Table 7: LLM Reliability Evaluation Results. Performance of candidate LLMs on reliability metrics (AvgSD ↓, Krippendorff's Alpha ↑, Fleiss' Kappa ↑) used to inform judge selection.

Model	AvgSD ↓	Krippendorff's Alpha ↑	Fleiss' Kappa ↑
`Claude-3.7`	0.280	0.634	0.446
`Gemini-2.0-flash`	0.147	0.669	0.654
`Gemma-3-27b`	0.191	0.732	0.562
`GPT-4o`	0.348	0.740	0.454
`Llama-3.3-70b-instruct`	0.276	0.669	0.541
`DeepSeek-V3-0324`	0.365	0.536	0.447
`Llama-4-maverick`	0.108	0.601	0.541
`Mistral-small-3.1`	0.411	0.376	0.148
`Llama-3.3-nemotron-49b`	0.543	0.182	0.085

D Entity and Attribute Filtering Mechanism

To ensure the quality and suitability of entities and attributes for synthesizing comparison multi-hop questions, we employ a filtering mechanism based on evaluating the concreteness of subject entities and the comparability of their attribute values. This process assigns numerical scores on a 1-5 scale, facilitating downstream filtering of less ideal candidates. The detailed criteria, as defined in

our COMPARE_ENTITY_FILTER_PROMPT, are summarized below:

D.1 Subject Entity Concreteness evaluation

The concreteness_score evaluates how specific, tangible, and suitable an entity is for direct attribute comparison. The scale is defined as:

- **5 (Highly Concrete):** Specific person, place, organization, tangible object, work, or clearly defined historical event (e.g., "Paris", "IBM"). Excellent candidate.
- **4 (Concrete):** Specific but less common entity types, like a specific named award or law (e.g., "Nobel Prize in Physics"). Good candidate.
- **3 (Borderline/Slightly Abstract):** Broader but well-defined categories or specific complex relationships (e.g., "Mammal", "World War II"). Use with caution.
- **2 (Abstract):** General relationships, abstract concepts, fields of study (e.g., "US-China relations", "Democracy"). Poor candidate.
- **1 (Highly Abstract):** Vague concepts, general feelings, ambiguous terms (e.g., "Happiness"). Unsuitable candidate.

D.2 Attribute Comparability Evaluation

The comparability_score evaluates each attribute value based on its suitability for direct and unambiguous comparison. The scale is defined as:

- **5 (Excellent):** Precise Dates (YYYY-MM-DD), specific Years, specific Numbers, exact Locations, specific Names, well-defined unambiguous Categories (e.g., Nationality).
- **4 (Good):** Specific but slightly less precise numbers (e.g., "1.2 million"), specific office/rank titles.
- **3 (Fair):** Broader categories, precise year ranges, specific event names. Potential candidate but less ideal.
- **2 (Poor):** Imprecise time (e.g., "Before 1960s"), descriptive reasons, lists. Unlikely suitable.
- **1 (Very Poor):**Vague statements, subjective opinions, long text. Unsuitable.

The filtering module processes each entity and its attributes based on these scoring criteria. Entities and attributes that do not meet a predefined threshold (default setting: a minimum score of 5 for subject entities and 4 for attributes) are filtered out before proceeding to the comparison query generation step (see Section 3.2 for their position in the pipeline). This specific 5/4 threshold was adopted based on the evaluation standards of a particular open-source model (Gemma-3-27B-it, as detailed in Appendix E), making it a reasonable choice for our study. Researchers can adjust it to fit different models or specific task requirements. This ensures that only entities with a sufficient level of concreteness and attributes with high comparability are used for synthesizing comparison questions, thereby enhancing the quality and relevance of the synthesized questions. The output format for these scores is a delimited string, with the first part being the entity's concreteness score and subsequent parts detailing each attribute's name, value, and comparability score.

E Experimental Settings

Corpus. We use the English Wikipedia dump from December 20, 2018. This date was chosen to align closely with the Wikipedia snapshots used in the baseline datasets (HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (Ho et al., 2020), and MusiQue (Trivedi et al., 2022)), ensuring a fair comparison of synthesized question quality under consistent data conditions. We preprocess the corpus by removing articles with more than 4096 words and store the remaining articles in JSONL format.

Generator LLMs. We employ four LLMs with varying capabilities and parameter sizes for question synthesis pipeline: Gemini-2.5-flash-preview-04-17, QwQ-32B, Qwen-14B, and GLM-4-9B-0414. Preliminary experiments showed that older or smaller models often lacked the capability to perform the complex generation steps. To prevent potential **self-enhancement bias** (Ferrara, 2023), these generator LLMs are deliberately kept distinct from those LLMs that constitute our LLM-as-judge evaluation system.

We also note that the outputs of closed-source models can fluctuate over time, potentially impacting reproducibility, an effect we observed with the Gemini model during our experimental period.

LLM-as-judge Models. Detailed in Appendix C.3.

Embedding and Reranker Models. For initial retrieval (coarse retrieval and MMR calculation) during the synthesis pipeline, we uniformly use the `gte-multilingual-base` embedding model. For the reranking stage (Step 2 in Section 3.1), we use `BAAI/bge-reranker-v2-m3`. For the retrieval-based dataset Evaluation (Section 4.3), we evaluate using `gte-multilingual-base`, `E5-base-4k`, and BM25. Our retrieval implementation is based on the FlashRAG (Jin et al., 2024).

Parameters for Coarse Retrieval In the coarse retrieval stage for bridge question synthesis (Section 3.1, Step 2), the MMR-like scoring function utilizes three trade-off parameters. We empirically set these parameters as follows: $\lambda_1 = 0.87$ (emphasizing query relevance), $\lambda_2 = 0.03$ (penalizing similarity to the source document), and $\lambda_3 = 0.1$ (promoting diversity among selected documents). These values were determined through preliminary experiments to balance the objectives of relevance, novelty, and diversity in the retrieved complementary documents.

Comparison Datasets. We compare the quality of HopWeaver-synthesized questions against three established human-annotated multi-hop QA datasets: HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (Ho et al., 2020), and MusiQue (Trivedi et al., 2022)

Evaluation LLM. For the QA-based dataset Evaluation (Section 4.2), we use `GPT-4o`, `Gemini-2.0-flash`, `Qwen3-9B`, `Llama-3.3-70B` to obtain answers from LLMs with different performance levels.

Polisher LLM. For the Polisher module experiments (Section 4.2), we use `DeepSeek-R1` to get an effective supervision signal.

Filter LLM. For the Filter module experiments (Step 2 in Section 3.2), we use `Gemma-3-27B-it` to get a stable rating threshold, which avoids fluctuations in question quality caused by different LLMs' standards when picking up entities and attributes.

Default Generation Parameters. For all LLMs in our experiments, we used deterministic generation settings (`temperature=0`, `do_sample=False`) with `top_p=0.9` and `max_tokens=8192` to ensure reproducibility while accommodating complex reasoning chains required for multi-hop question synthesis.

F Prompt Settings

Considering that this work involves a substantial number of prompts, each with a relatively lengthy original format, we have made appropriate simplifications while retaining the essential information from the original prompts. Readers interested in the complete original prompts should refer to the code file provided.

Bridge Entity Extraction Prompt (ENTITY_EXTRACTION_PROMPT)

Goal

Given a text document, select a single segment with high potential to contain a bridge entity for multi-hop question generation, identify one bridge entity from that segment, extract relevant text segments, and generate an expanded query statement for this bridge entity to retrieve related documents from a vector database.

Instructions

1. **Select a Segment and Identify a Bridge Entity** - Select a text segment with high potential for containing a bridge entity, then identify one bridge entity. - **Principles**: - **High Connectivity**: The entity has multiple associations with other entities. - **Uniqueness and Clarity**: The entity is clearly defined within the segment. - **Attribute Richness**: The entity has multiple queryable attributes. - **Cross-Document Distribution**: Information likely spread across documents. - **Distinct from Title**: The bridge entity must not be identical to the document title. - Format: `("bridge_entity" <|> "entity_name" <|> "entity_type")`
2. **Extract Relevant Text Segments** - Extract a single part of the document that directly mentions or describes the entity. - Provide a brief introductory sentence followed by the extracted segment. - Format: `("relevant_segments" <|> "entity_name" <|> "entity_introduction + extracted_part")`
3. **Generate an Expanded Query Statement** - Generate a query to find COMPLEMENTARY information about the entity. - Use semantic direction shifting phrases like "instead of," "beyond," etc. - Format: `("query" <|> "entity_name" <|> "entity_query")`
4. **Return Output** - Return a single list with the bridge entity, relevant segments, and expanded query.
5. **When Finished** - Output `<|COMPLETE|>`

Sub-Question Generation Prompt (SUB_QUESTION_GENERATION_PROMPT)

Goal

Analyze two documents connected by a bridge entity and generate two sequential sub-questions that form a multi-hop reasoning chain.

Instructions

Analyze how the bridge entity connects both documents by:
- Identifying key information about the bridge entity in Document A that is unique to Document A.
- Finding related information in Document B that connects via this bridge entity.
- Determining a clear reasoning path from Document A to Document B.
- If no valid bridge connection exists, return `INVALID_BRIDGE_CONNECTION` with explanation.

Generate two sequential sub-questions:
- **Sub-question 1**: A question about Document A where the answer is the bridge entity.
- **Sub-question 2**: A question that explicitly uses the bridge entity to find related information in Document B.

Each sub-question must:
- Be answerable from only one document.
- Have a definitive answer contained in its document.
- Be phrased as a standalone question without document references.
- Be specific with clear references to information in its document.
- Provide a clear, concise answer.
- Together form a logical reasoning chain.

Output Format

If no valid bridge connection exists:

```
INVALID_BRIDGE_CONNECTION
Reason: [Brief explanation]
```

If valid bridge connection exists:

```
ANALYSIS:
Bridge connection: [How the bridge entity connects the documents]
Document A segments: [Copy of the original Document A segments]
Document B segments: [Relevant excerpts from Document B]
Reasoning path: [Logical path from Document A to Document B]

SUB-QUESTIONS:
Sub-question 1: [Question about Document A]
Answer 1: [Answer from Document A - about the bridge entity]

Sub-question 2: [Question using bridge entity to find answer in Document B]
Answer 2: [Answer from Document B]
```

Multi-Hop Question Synthesis Prompt (MULTI_HOP_QUESTION_SYNTHESIS_PROMPT)

Goal

Synthesize a concise, natural multi-hop question that requires reasoning across two documents, connecting two sub-questions into a single logical inquiry.

Instructions

- FIRST, check if the bridge entity (Answer 1 from the first sub-question) is included in the text of the second sub-question. If not, return `NONE`.
- Review the analysis and sub-questions to trace the full reasoning chain.
- Create a single multi-hop question that:
- Is ONE cohesive question, not multiple questions combined
- Requires distinct information from both Document A and B
- Reads naturally as a coherent, conversational question
- Cannot be fully answered using only one document
- Follows the reasoning path of the sub-questions, using the bridge entity (Answer 1) to link to information in Document B
- Is clear, concise, and free of ambiguity
- Doesn't explicitly mention the bridge entity or reasoning steps
- If the sub-questions cannot be combined into a valid multi-hop question, return `NONE` with explanation.
- Ensure the final answer matches Answer 2 (the answer from Document B) from the sub-questions.

Output Format

If sub-questions cannot be combined:

```
NONE
Reason: [Brief explanation]
```

If a valid multi-hop question can be created:

```
MULTI-HOP QUESTION: [Your synthesized question]

ANSWER:
[The final answer, matching Answer 2 from Document B]

REASONING PATH:
[Step-by-step explanation showing:
 1. How to find the bridge entity (Answer 1) in Document A
 2. How this bridge entity leads to the final answer in Document B]

SOURCES:
[Document A and Document B, specifying their roles]
```

Bridge Polisher Prompt (POLISHER_PROMPT)

Goal

Validate and refine multi-hop questions to ensure they genuinely require cross-document reasoning and follow a proper reasoning chain where information from one document is essential to answer a question about content in another document.

Instructions

You are a Polisher module responsible for validating and refining multi-hop questions. Given a multi-hop question, its suggested answer, reasoning path, and source document segments, you will evaluate the question's quality and make one of four decisions:

1. **PASS**: The question is valid, well-formed, and genuinely requires both documents.
2. **ADJUST**: The question needs surface wording improvements only.
3. **REWORKED**: The question needs substantial structural changes.
4. **REJECTED**: The question has unfixable flaws.

Review and modify the question based on these key dimensions:

1. **True Multi-hop Necessity**: CRITICAL
- Information must flow from Document A to Document B in a logical sequence
- The answer must be impossible to determine using either document in isolation
- The reasoning path must demonstrate how Document A provides necessary context
- The question should require discovering connections not explicitly stated

2. **Hidden Bridge Structure**:
- The question should NOT directly mention the connecting entity or concept
- The bridge entity should remain implicit in the question wording
- The question should require identifying the relevant bridge entity
- Reframe questions that explicitly name the bridge entity

3. **Reasoning and Answer Quality**:
- Verify the reasoning follows a logical progression between documents
- Ensure the answer is factually accurate according to both documents
- Check that the answer requires synthesizing information across documents
- Improve question wording for clarity, fluency, and natural tone

Output Formats

1. If the question passes all criteria without changes:

```
[PASS]
```

2. If the question needs minor adjustments:

```
[ADJUST]
REFINED_REASONING_PATH: [Updated reasoning path]
REFINED_QUESTION: [Adjusted question]
REFINED_ANSWER: [Updated answer if needed]
```

3. If the question needs significant refinement:

```
[REWORKED]
REFINED_REASONING_PATH: [Revised reasoning path]
REFINED_QUESTION: [Substantially revised question]
REFINED_ANSWER: [Updated answer]
```

4. If the question is fundamentally flawed:

```
[REJECTED]
```

Bridge MHQA Quality Assessment Prompt (MHQA_QUALITY_ASSESSMENT_PROMPT)

Goal
Conduct a **rigorous and critical** evaluation of multi-hop questions and their answers across multiple quality dimensions. Focus on ensuring questions require genuine cross-document reasoning **and are free from logical flaws**. A high-quality multi-hop question necessitates reasoning that flows between documents, where information from one document provides context for another, and the answer must be impossible to determine using any single document in isolation.
Instructions
You are a **strict and discerning** Multi-Hop Question Answering (MHQA) dataset quality assessment expert. Evaluate the given multi-hop question and its answer across key dimensions in three categories. **Apply rigorous scrutiny and do not hesitate to assign lower ratings if flaws are present, especially logical ones.**
1. **Multi-Hop QA Rule Dimension** - **Multi-Hop Reasoning Requirement**: Does the question genuinely require reasoning across multiple documents? (Yes/No)
2. **Linguistic Dimensions** (Rate as: Very Poor, Poor, Fair, Good, Very Good) - **Fluency**: Is the question grammatically correct, coherent, and easy to understand? - **Clarity**: Is the question clearly and precisely expressed without ambiguity? - **Conciseness**: Is the question concise without redundant information?
3. **Task-Oriented Dimensions** (Rate as: Very Poor, Poor, Fair, Good, Very Good) - **Relevance**: Is the question relevant to the given passages and asking for key information? - **Consistency**: Is the information in the question **completely and strictly** consistent with the provided passages? - **Question Answerability**: Can the question be **unambiguously** answered based **solely** on the given passages? - **Answer-Question Consistency**: Does the provided answer completely and accurately address the question? - **Information Integration Ability**: Does the question successfully integrate information from multiple documents **without forcing unnatural connections**? - **Reasoning Path Guidance**: Does the question guide the answerer through a multi-step reasoning process? - **Logical Sophistication**: Does the question demonstrate clever design that requires multi-step thinking and is **free from logical gaps or fallacies**?
Critical Scoring Guidance: - **Penalize Logical Flaws Heavily:** Pay close attention to Consistency, Logical Sophistication, and Information Integration Ability. - **Multi-Hop Requirement is Paramount:** If this requirement is "No," the question fundamentally fails. - **Clarification on 'Fair':** A 'Fair' rating signifies only basic adequacy and is not a positive endorsement.
Rating Scale Interpretation: - **Very Poor:** Unacceptable quality with serious functional/logical errors - **Poor:** Weak/Barely Usable quality with obvious, major flaws - **Fair:** Acceptable/Passable quality meeting basic requirements with clear flaws - **Good:** Standard good quality, well-designed without obvious flaws - **Very Good:** Excellent/Outstanding quality with clever, rigorous design
Output Format:

```
- Multi-Hop Reasoning Requirement: {yes/no}
- Fluency: {rating}
- Clarity: {rating}
- Conciseness: {rating}
- Relevance: {rating}
- Consistency: {rating}
- Question Answerability: {rating}
- Answer-Question Consistency: {rating}
- Information Integration Ability: {rating}
- Reasoning Path Guidance: {rating}
- Logical Sophistication: {rating}
<|COMPLETE|>
```

Compare Entity Extraction Prompt (COMPARE_ENTITY_EXTRACTION_PROMPT)

Goal

Given a text document, identify its primary subject entity and extract multiple key attributes associated with this entity, along with their corresponding values. For each extracted attribute, generate an expanded query statement designed to retrieve documents about *other* similar entities that also possess this attribute, facilitating subsequent comparison.

Instructions

1. **Identify the Primary Subject Entity**

- Determine the main person, place, organization, event, concept, or work that the document is primarily about.
- Determine the **subject_entity_name** (capitalized) and its general **subject_entity_type**.

2. **Extract Comparable Attributes, Values, and Generate Queries**

- Identify **multiple (aim for 3-5 if possible)** distinct attributes associated with the primary subject entity.
- **Focus strictly on attributes whose VALUES are suitable for comparison.** Prioritize attributes that meet these criteria:
- **Concise & Factual Value:** Short value (e.g., name, number, date, category, location).
- **Common Data Types:** Prefer Numbers, Dates, Locations, Specific Names, Defined Categories.
- **Likely Commonality:** Prefer attributes likely to exist for other similar entities.
- **For each identified comparable attribute:**
- Determine the **attribute_name** (e.g., "Population", "Date of Birth").
- Extract the **attribute_value** (e.g., "1.2 million", "1990-05-15").
- Generate an **entity_b_query**: A concise query to find *other* entities with the same attribute.

3. **Output Format Specification**

- **Subject Entity Part:** `("subject_entity"<|>"subject_entity_name"<|>"subject_entity_type")`
- **Attribute Parts:** `("attribute"<|>"attribute_name"<|>"attribute_value"<|>"entity_b_query")`
- Use `##` as delimiter between parts.
- Append `<|COMPLETE|>` at the end.

Compare Entity Filter Prompt (COMPARE_ENTITY_FILTER_PROMPT)

Goal

Assess the concreteness of a pre-identified subject entity and the comparability of its extracted attribute values. Assign numerical scores reflecting these assessments on a 1-5 scale to facilitate downstream filtering.

Instructions

1. **Assess Subject Entity Concreteness:** - Evaluate the provided `subject_entity_name` and `subject_entity_type`. - Assign a **`concreteness_score`** on a scale of 1 to 5:

- **5 (Highly Concrete):** Specific person, place, organization, tangible object, work, or defined event. (e.g., "Mihály Mosonyi", "Paris", "IBM")
- **4 (Concrete):** Specific but less common entity types. (e.g., "Nobel Prize in Physics", "Treaty of Versailles")
- **3 (Borderline/Slightly Abstract):** Broader well-defined categories. (e.g., "Mammal", "Impressionism")
- **2 (Abstract):** General relationships, abstract concepts, fields of study. (e.g., "US-China relations", "Democracy")
- **1 (Highly Abstract):** Very vague concepts, feelings, ambiguous terms. (e.g., "Happiness", "The problem with X")

2. **Assess Attribute Comparability:** - Evaluate **each** provided `attribute_value`. - Assign a **`comparability_score`** (scale 1-5):

- **5 (Excellent):** Precise Dates, Years, Numbers, exact Locations, specific Names, well-defined Categories.
- **4 (Good):** Specific but slightly less precise numbers, specific titles.
- **3 (Fair):** Broader categories, year ranges if precise, specific event names.
- **2 (Poor):** Imprecise time, descriptive reasons, lists.
- **1 (Very Poor):** Vague statements, subjective opinions, long text.

3. **Format Output:** Generate the output string according to the specification.

Output Format Specification

Strictly adhere to the following output format:

1. **Structure:** The entire output must be a single string containing multiple parts delimited by `##`.
2. **First Part (Entity Score):** The *first* part MUST represent the entity's concreteness score. Format: `("entity_score"<|>5)` (example shows score of 5).
3. **Subsequent Parts (Attribute Scores):** For *each* attribute provided in the input, include a corresponding scoring part. Format: `("attribute_score"<|>"Birth Date"<|>"4 September 1815"<|>5)` (example shows score of 5 for a date attribute).
4. **Delimiter:** Use `##` strictly as the delimiter *between* parts. Do not use it at the beginning or end.
5. **Completion Signal:** Append `<|COMPLETE|>` to the very end of the entire generated string.

Comparison Polisher Prompt (COMPARISON_POLISHER_PROMPT)

Goal

Validate and optimize **comparison-type** questions to ensure correct comparison logic, clear and natural phrasing, and sufficient background information to enhance question quality and comprehensibility.

Instructions

You are a Polisher module responsible for optimizing comparison questions. Based on the input of two entities (A and B), the attribute being compared, supporting facts, the original question-answer pair, and relevant document contexts, evaluate the quality of the question and make one of the following four decisions:

1. **PASS**: The question is valid, well-phrased, has correct comparison logic, and appropriate background information.
2. **ADJUST**: The question is basically valid but needs fine-tuning in wording, fluency, or background information.
3. **REWORKED**: The question has obvious flaws and needs structural rewriting.
4. **REJECTED**: The question has fundamental errors that cannot be fixed.

Review and modify the question based on the following key dimensions:

1. **Comparison Correctness (CRITICAL)**: - **Attribute Comparability**: Confirm that the attributes of entities A and B are indeed comparable. - **Logical Accuracy**: Verify that the comparison logic is consistent with the values provided. - **Answer Consistency**: Ensure the original answer accurately answers the question. - **Factual Support**: Check that the facts are key information extracted from the documents.
2. **Background Information Integration (IMPORTANT)**: - **Natural Integration**: Extract key background information and integrate it naturally. - **Provide Context Without Revealing Answers**: Background should provide context without revealing attribute values. - **Context Relevance**: Added background information should be relevant to the entities and attributes.
3. **Question Wording Optimization**: - **Clarity and Naturalness**: Improve wording to make it clear, fluid, and conversational. - **Direct Comparison Format**: Ensure the question explicitly asks for the result of the comparison. - **Hide Answer-Revealing Details**: Never include specific attribute values that would reveal the answer. - **Unified Question Format**: Create a single, unified question with smooth background incorporation.

Output Format

1. If the question needs no modification:

```
[PASS]
```

2. If the question needs fine-tuning:

```
[ADJUST]
REFINED_QUESTION: [Unified question with background]
REFINED_ANSWER: [Adjusted answer if needed]
```

3. If the question needs substantial rewriting:

```
[REWORKED]
REFINED_QUESTION: [Completely rewritten question]
REFINED_ANSWER: [New answer]
REFINED_FACT_A: [Corrected fact for entity A if needed]
REFINED_FACT_B: [Corrected fact for entity B if needed]
```

4. If the question cannot be fixed:

```
[REJECTED]
REASON: [Brief explanation of rejection reason]
```

Compare Question Builder Prompt (COMPARE_QUESTION_BUILDER_PROMPT)

Goal
Imagine you are comparing two documents, Document A (about Entity A) and Document B (a candidate potentially containing a related Entity B). Your task is to:

1. Identify the main subject entity within Document B (potential Entity B) and see if it's relevant to Entity A.
2. Find if there is **at least one specific, comparable attribute pair** between Entity A and the potential Entity B.
3. If a suitable comparison pair is found, **directly generate** a natural language **direct comparison question**, its **comparative answer**, and supporting **full sentence(s)**.
4. If no suitable entity or comparable attribute pair is found, indicate failure.

Instructions
1. **Analyze Inputs:** You are given:
- Primary Entity A: **{subject_entity_name}** (Type: **{subject_entity_type}**)
- Document A Text: **{document_a_text}**
- Entity A's Attributes List: **{attributes_list_str_a}**
- Candidate Document B Text: **{document_b_text}**
2. **Identify Entity B and Find ONE Comparable Attribute Pair:**
- Identify the primary subject entity within Document B (Entity B).
- Check if Entity B's type is compatible for comparison with Entity A's type.
- Search for a comparable attribute pair between Entity A and Entity B.
- If found, proceed to step 3. Otherwise, proceed to step 4 (Failure).
3. **Generate DIRECT Comparison Question, COMPARATIVE Answer, and Facts:**
- Compare values to determine their precise relationship.
- Generate a **direct comparison question** that explicitly asks for the result of comparison.
- Provide a **concise comparative answer** (not just restating both values).
- Extract supporting sentences from both documents.
- Extract relevant paragraphs (50-150 words) from both documents.
4. **Indicate Failure** if no comparable pair or valid Entity B found.
Output Format Specification
1. Success Output (If a comparable pair was found):

```
PASS
entity_a: Name of Entity A (from input)
entity_b: Identified Entity B Name (from step 2)
attribute_compared: Matched Attribute Name
multi_hop_question: Generated DIRECT Comparison Question
answer: Concise COMPARATIVE Answer Text
fact_entity_a: Extracted Full Sentence(s) for Fact A
fact_entity_b: Extracted Full Sentence(s) for Fact B
relevant_paragraph_a: Complete substantive paragraph from Document A
relevant_paragraph_b: Complete substantive paragraph from Document B
```

2. Failure Output:

```
FAIL
```

Compare Query Generator Prompt (COMPARE_QUERY_GENERATOR_PROMPT)

Goal

Imagine you are an assistant helping to create interesting comparison questions that might require looking up information in different places (multi-hop). Your task is to analyze a primary entity (Entity A) and its known details. Based on this, decide the best *first step* to find another entity (Entity B) for comparison: either confidently suggest a specific Entity B and verify a *known attribute* of Entity A for it, OR generate 3 diverse search queries to explore potential candidates.

Instructions

1. **Analyze Input Context:** You are working with:
- Primary Entity A: **{subject_entity_name}** (Type: **{subject_entity_type}**)
- Context about Entity A: **{document_a_text}**
- Known Attributes of Entity A: **{attributes_list_str_a}**

2. **Consider Two Paths (Choose ONE):**

Path 1: Direct Entity Recall & Focused Verification Query:
- **Think:** Based on Entity A's profile, can you confidently recall a *specific* entity (Entity B) that's relevant?
- Identify **one specific attribute** (Attribute X) *from the provided list* that would be an interesting point of comparison.
- **Condition:** Choose this path ONLY if you can confidently recall Entity B *and* select a suitable Attribute X.
- **Generate Verification Query:** Create a query to retrieve the value of that chosen attribute for Entity B.
- **Output:** Format as `("recall_focused_verify"<|>[Entity B Name]<|>[Attribute X Name]<|>[Verification Query])`.

Path 2: Heuristic Search Query Generation:
- **Think:** If Path 1 isn't suitable, generate search queries to explore potential entities.
- **Generate Exactly 3 Queries:** Propose diverse search queries based on Entity A's overall profile.
- Ensure queries are concise, suitable for retrieval, and explore different angles.
- **Condition:** Choose this path if Path 1 is not suitable.
- **Output:** Format as `("search_queries"<|>Query 1<|>Query 2<|>Query 3)`.

3. **Output:** Return the chosen path's output string. You must choose exactly one path.

Output Format Specification

1. Structure: A single string containing the chosen path information.

2. Output Parts (Choose ONE format):

```
Path 1: ("recall_focused_verify"<|>Suggested Entity B Name<|>
Chosen Attribute X Name<|>Verification Query)
Path 2: ("search_queries"<|>Query 1<|>Query 2<|>Query 3)
```

3. Completion Signal: Append `<|COMPLETE|>` at the end.

Compare QA Quality Assessment Prompt (COMPARE_QA_QUALITY_ASSESSMENT_PROMPT)

Goal

Conduct a **rigorous and critical** evaluation of multi-hop comparison questions across multiple quality dimensions. Focus on ensuring questions require genuine cross-document reasoning **and are free from logical flaws**. A high-quality multi-hop question necessitates reasoning that flows between documents, where information from one document provides necessary context for another, making it impossible to answer using any single document in isolation.

Instructions

You are a **strict and discerning** Multi-Hop Question Answering (MHQA) expert evaluating the given question and answer across key dimensions in three categories:

1. **Multi-Hop QA Rule Dimension**

- **Multi-Hop Reasoning Requirement**: For comparison-type questions, determine if:

1) Answering requires factual information from at least two different documents

2) No single document contains all necessary information about both entities being compared

Rate "Yes" only if BOTH conditions are met, otherwise "No"

2. **Linguistic Dimensions** (Rate as: Very Poor, Poor, Fair, Good, Very Good)

- **Fluency**: Is the question grammatically correct, coherent, and easy to understand?
- **Clarity**: Is the question clearly and precisely expressed without ambiguity?
- **Conciseness**: Is the question concise without redundant information?

3. **Task-Oriented Dimensions** (Rate as: Very Poor, Poor, Fair, Good, Very Good)

- **Relevance**: Is the question relevant to the passages and asking for key information?
- **Consistency**: Is the question **completely and strictly** consistent with the passages?
- **Question Answerability**: Can the question be **unambiguously** answered based **solely** on the passages?
- **Answer-Question Consistency**: Does the answer completely and accurately address the question?
- **Information Integration**: Does the question logically integrate information from multiple documents?
- **Reasoning Path Guidance**: Does the question guide the answerer through a multi-step reasoning process?
- **Logical Sophistication**: Is the question free from logical gaps and requires multi-step thinking?

Critical Scoring Guidance:

- **Penalize Logical Flaws Heavily**: Pay close attention to Consistency, Logical Sophistication, and Information Integration.
- **Multi-Hop Requirement is Paramount**: If "No," the question fundamentally fails its purpose.
- **Rating Scale Interpretation**:
- **Very Poor**: Unacceptable quality with serious functional/logical errors
- **Poor**: Weak quality with obvious, major flaws requiring significant revision
- **Fair**: Acceptable quality meeting basic requirements with clear flaws (minimum adequacy only)
- **Good**: Standard good quality, well-designed without obvious flaws
- **Very Good**: Excellent quality with clever, rigorous design and deep logic

Output Format:

```
- Multi-Hop Reasoning Requirement: {yes/no}
- Fluency: {rating}
- Clarity: {rating}
. . .
- Reasoning Path Guidance: {rating}
- Logical Sophistication: {rating}
<|COMPLETE|>
```