MoNaCo: More Natural and Complex Questions for Reasoning Across Dozens of Documents

Tomer Wolfson[1], **Harsh Trivedi**[2], **Mor Geva**[3], **Yoav Goldberg**[2,4],
Dan Roth[1,5], **Tushar Khot**[*], **Ashish Sabharwal**[2], **Reut Tsarfaty**[4]
[1]University of Pennsylvania, USA [2]Allen Institute for AI, USA [3]Tel Aviv University, Israel
[4]Bar-Ilan University, Israel [5]Oracle AI, USA
wolfsont@seas.upenn.edu

Abstract

Automated agents, powered by large language models (LLMs), are emerging as the go-to tool for querying information. However, evaluation benchmarks for LLM agents rarely feature natural questions that are both information-seeking and genuinely time-consuming for humans. To address this gap we introduce MoNaCo, a benchmark of 1,315 natural and time-consuming questions that require dozens, and at times hundreds, of intermediate steps to solve—far more than any existing QA benchmark. To build MoNaCo, we developed a decomposed annotation pipeline to elicit and manually answer real-world time-consuming questions at scale. Frontier LLMs evaluated on MoNaCo achieve at most 61.2% F1, hampered by low recall and hallucinations. Our results underscore the limitations of LLM-powered agents in handling the complexity and sheer breadth of real-world information-seeking tasks—with MoNaCo providing an effective resource for tracking such progress. The MoNaCo benchmark, codebase, prompts, and models predictions are all publicly available at: `https://tomerwolgithub.github.io/monaco`.

1 Introduction

Large language models (LLMs), and the ''agentic'' systems built around them, are becoming increasingly ingrained into how people seek and query information (Guo et al., 2024; OpenAI, 2025a). It is therefore important to understand how well LLM-powered agents fare in answering real-world questions that require combining information from dozens or even hundreds of documents. For example, a political scientist may want to know whether ''*In European countries, are left-wing political parties more likely to be headed by women than right-wing ones?*'' a question which, unless answered verbatim in some source, is extremely time-consuming. In fact, answering this question entails reviewing all current political parties in each European country, extracting their political affiliation and leader's gender, and then to reason over all these intermediate facts, combining results from 719 distinct pages (see Table 1).

While LLM-powered agents hold great promise in solving realistic time-consuming tasks, such questions are not well represented in contemporary QA benchmarks (Wei et al., 2024, 2025). Building a benchmark that contains challenging questions which are *also* natural is no easy feat. QA benchmarks that focus on *natural* questions typically contain simple questions, answerable using a single passage of text (Abujabal et al., 2019; Kwiatkowski et al., 2019). Collecting questions that are realistic and *complex* has largely been reserved for domain experts, which incur high annotation costs and often result in benchmarks that are small or have questions that are not particularity time-consuming, involving only a handful of documents (Malaviya et al., 2024; Yoran et al., 2024a). As it is non-trivial to collect complex natural questions ''in the wild'', researchers have instead focused on creating such benchmarks artificially (Trivedi et al., 2022; Li et al., 2024; Wei et al., 2025). However, machine-based approaches often result in contrived questions that do not reflect real-world users' needs (x-axis of Figure 1).

To address this gap we introduce **MoNaCo**, a benchmark of **Mo**re **Na**tural and much more **Co**mplex questions. This benchmark is designed to evaluate LLM-based systems on information-seeking tasks that are realistic and time-consuming, and demand planning, collecting, and synthesizing many of pieces of information.

[*]Work done while the author was at Allen AI.

Transactions of the Association for Computational Linguistics, vol. 14, pp. 23–46, 2026. https://doi.org/10.1162/TACL.a.64
Action Editor: Benjamin Van Durme. Submission batch: 7/2025; Revision batch: 9/2025; Published 1/2026.

Domain	Example	# Pages	# Steps
Politics	In European countries, are current left-wing political parties more likely to be headed by women than right-wing ones?	719	16
History	What has been the highest percentage of parliament seats held by monarchist parties during the time of the Third French Republic?	49	10
Demographics	Which Nobel Prize category has the fewest number of Asian-born recipients?	871	6
Sports	What was the youngest team in the NBA in 2021?	550	5
Culinary	I want to cook a traditional Bulgarian meal for my girlfriend, that is allergic to eggs and dairy, which dishes would be okay?	42	3
Art	Which museums house the most famous paintings by each of the leading French Impressionists?	58	3
Literature	Which books by Gabriel Garcia Marquez are based on real historical events?	21	3
Music	What percentage of musicians with Billboard Year-end Hot 100 singles were born outside the United States in 2020, 2010 and 2000?	449	22
Film & TV	What are the 3 most common professions among the fathers of Oscar winners for best actress?	75	5
Pop Culture	What are the most common names for girls in the UK that are not originally Biblical?	21	3
Other	What percentage of US supreme court justices throughout history did not attend an Ivy League school for their postgraduate degree?	59	7

Table 1: Questions from MoNaCo, along with their number of unique Wikipedia pages containing the necessary evidence (# Pages), and the number of decomposition steps required to answer each question (# Steps).

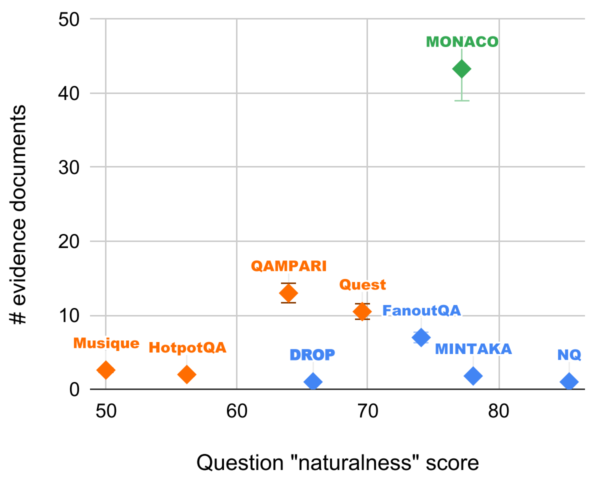

Figure 1: MoNaCo (in green) and 8 QA benchmarks, plotted based on their ‘‘naturalness’’ (judged by humans) and their number of evidence documents. Natural QA benchmarks are in blue, artificial ones in orange. MoNaCo targets the top-right region: natural questions that require dozens of evidence documents.

MoNaCo contains 1,315 challenging multi-step questions whose solution involves retrieving, filtering, and aggregating dozens of intermediate facts, found in both unstructured and structured sources (paragraphs and tables). All questions in MoNaCo come with a gold-standard, manually annotated reasoning chain that contains all of the intermediate steps, answers and supporting evidence required to solve it. Namely, each question has between 1 and 2,379 Wikipedia paragraphs and tables that are needed to solve it—with an average of 43.3 unique pages per question. The questions are designed to reflect the information-seeking goals of human personas (history professor, amateur chef, etc.), with a special emphasis on time-consuming tasks (Table 1).

To construct MoNaCo, we first developed an approach for eliciting complex questions from humans. Instead of relying on pre-defined templates, we prompted workers to generate questions that would engage specific ‘‘target personas’’. This method produced questions that are not only more challenging but also perceived as more realistic, as confirmed by a user study comparing MoNaCo to 8 other complex QA benchmarks (§4). Concretely, on the x-axis of Figure 1, MoNaCo questions rank close to the natural yet simpler benchmarks (in blue), while being ranked much higher than all machine-generated benchmarks (in orange).

Collecting answers to MoNaCo is non-trivial as questions typically require combining information from dozens and even hundreds of documents. Therefore, we use the question decomposition method of Wolfson et al. (2020) to implement a *decomposed annotation pipeline*, breaking the process of annotating complex, time-consuming questions into multiple, simpler tasks (Figure 2). This distributed approach facilitates the annotation of gold answers by enabling non-expert workers to answer the simpler intermediate steps of much more challenging questions.

As an AI benchmark, MoNaCo enables us to evaluate LLM-based systems in 4 key settings: (1) evaluating parametric knowledge and reasoning, (2) evaluating complex reasoning over long contexts when all the information is provided, (3)

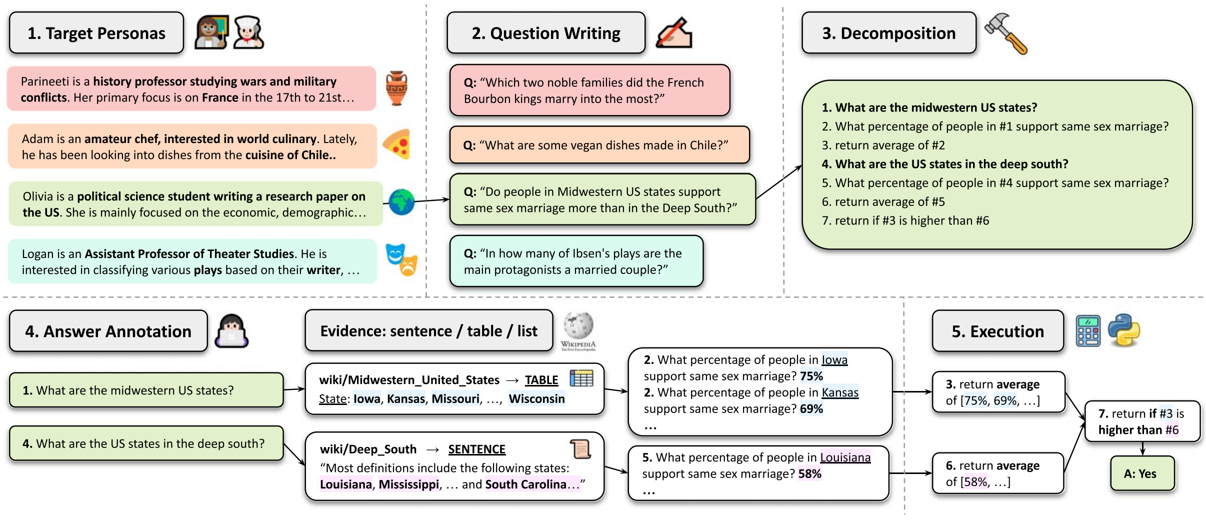

Figure 2: An overview of our annotation pipeline for building the MoNaCo benchmark.

evaluating end-to-end retrieval-augmented QA, and (4) evaluating multi-document retrieval. In our experiments (§5), we focus on the first three settings. First, we find that frontier LLMs, including recent AI reasoning models (OpenAI, 2024; DeepSeek AI, 2025), struggle on MoNaCo, with the top performing o3 model reaching an F1 score of 61.2%—with only 38.7% of the examples getting a perfect score. We observe that all models suffer from low recall and tend to hallucinate answers that involve aggregating many facts. Second, even in an Oracle retrieval setting, where all of the gold evidence is provided as context, GPT-4o and Llama 3.1-405B struggle to perform long-context reasoning, scoring only 59% F1. Third, retrieval-augmented generation did not help, with BM25 retrieval hurting LLMs' performance by over 12 points, exhibiting a lack of ''retrieval robustness'' (Yoran et al., 2024b).

Overall, our contribution is threefold:

- We design an annotation pipeline to elicit and annotate realistic, time-consuming questions, with answers that often span across dozens and even hundreds of documents (§2, §3).
- We release MoNaCo, a benchmark of 1,315 information-seeking questions that require retrieving and aggregating information from dozens of Wikipedia pages (§4).
- We experiment with 15 state-of-the-art LLMs and highlight the shortcomings of such models in answering MoNaCo questions (§5).

As LLM agents near human-level performance on a variety of tasks, it becomes increasingly important to expose their limitations, especially when tackling information-seeking questions. To this end, MoNaCo serves as a unique and challenging testbed for evaluating, LLM-powered, agentic and ''Deep Research'' systems (Huang et al., 2025) on broad tasks that span across potentially hundreds of documents, demanding extensive factual knowledge, information retrieval and reasoning skills.

2 Questions, Decompositions, & Answers

This section describes our novel approach for collecting complex natural questions, their corresponding answers and a comprehensive set of supporting evidence (Figure 2). Our first stage is *question writing*, where annotators write time-consuming questions that reflect the information-seeking goals of target personas (§2.1). In the next stage, we annotate each question with a formal *question decomposition* (§2.2). The decomposition enables us to facilitate the answer annotation by breaking down challenging questions into a series of much simpler annotation tasks. Finally, we outline our *decomposition execution engine*, which queries and aggregates intermediate answers in order to derive the final answer (§2.3).

2.1 Eliciting Complex Natural Questions

As discussed in §1, collecting realistic and time-consuming questions ''in the wild'' is hard: Users tend to shy away from issuing such queries

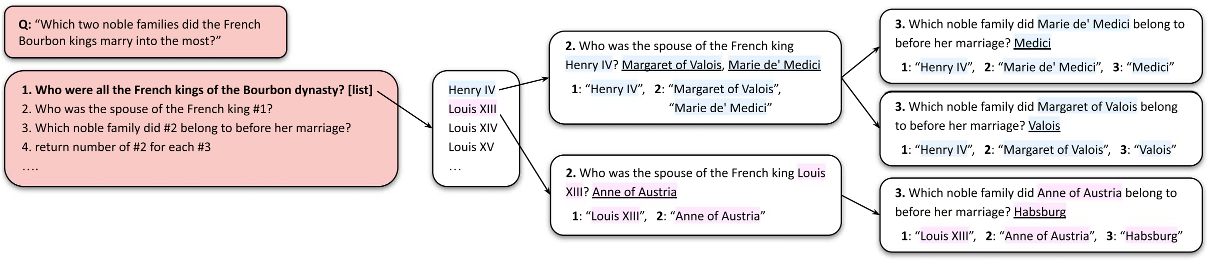

Figure 3: Iteratively deriving intermediate questions using the question decomposition. The intermediate answers from previous steps (highlighted) are assigned to the decomposition step, creating the follow-up questions. The full answer assignments deriving each question are also presented at the bottom of each question box.

to search engines (Kwiatkowski et al., 2019; Thakur et al., 2021), while collecting them from user-AI interactions, most of which are private, remains challenging (Lin et al., 2024). Instead, we rely on *human annotators* to write questions that reflect real-world users' information-seeking goals.

Our approach is to *prompt* crowd workers to write questions that would interest a particular *target persona*. To illustrate, Part 1 in Figure 2 presents four personas: a history professor, an amateur chef, a political scientist and a theater scholar. By priming workers to assume a specific persona and not to use any pre-defined question templates, we encourage them to come up with more realistic questions. In addition, the use of multiple distinct personas helps diversify our data. To encourage challenging questions, we included five randomly selected *reference questions* (initially written by us and later replaced by worker-written ones) which were intentionally time-consuming, often involving dozens of documents. The complete details of the question writing task are provided in Appendix B.1.

2.2 Annotating Intermediate Answers

Our next stage is to annotate each question with its corresponding answers and supporting evidence. Given that answering MoNaCo questions involves multiple steps and may require dozens of evidence documents, it is impractical for a single annotator to answer them directly. To facilitate answer annotation, we first have skilled workers annotate each question with its *question decomposition*, using QDMR (Wolfson et al., 2020) a widely used formalism for representing complex, multi-step questions (Saparina and Osokin, 2021; Geva et al., 2022; Wolfson et al., 2022).

The decomposition is a series of intermediate steps that form a plan for answering the original question.[1] Part 3 of Figure 2 displays the decomposition of the question *''Do people in Midwestern US states support same sex marriage more than in the Deep South?''*. By following the intermediate steps, workers are only asked to answer a series of simpler questions—while the derivation of follow-up questions and aggregation of intermediate answers is executed automatically (§2.3). As the answers to distinct intermediate questions are often found in separate sources, this also facilitates the annotation of questions whose evidence spans across many documents. For example, in Figure 2 the answers to steps 1 (*''Midwestern states''*) and 4 (*''states in the Deep South''*) lie in two separate documents, a table and a sentence, respectively.

2.3 Deriving the Final Answers

To query intermediate answers and derive the final answer, we implemented a *decomposition execution engine*. The role of the execution engine is twofold, as it leverages the inherent structure of the decomposition to map steps into either: (1) intermediate questions for annotation, or (2) executable Python programs that aggregate intermediate answers to produce the final answer.

First, the automatic derivation of intermediate questions, using the answers of previous steps, is presented in Figure 3. The execution engine assigns the annotated answers of step #1 (*Henry IV, Louis XIII, ...*) to step #2, thereby deriving its follow-up questions. Similarly, when deriving

[1] The same question may have multiple valid decompositions with varying levels of granularity, depending on the underlying information source. As we focus on Wikipedia, we made sure decompositions were annotated such that the answer to each intermediate question lies in a single page.

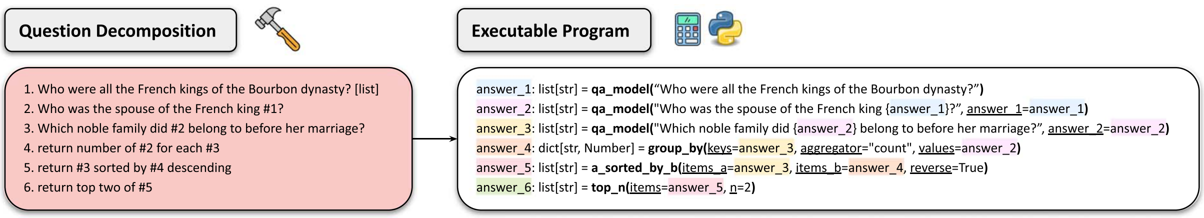

Figure 4: Executing a question decomposition as a Python program. Steps 1–3 have their answers annotated by crowd workers. Steps 4–6 represent query operators and their answers are automatically computed.

questions from step #3, referring to steps 1–2, it is assigned the answers from both these steps.

The second role of the execution engine is to query the answers of intermediate steps. Namely, decomposition steps can represent different query operators that filter or aggregate the answers of previous steps. Our execution engine provides a concrete implementation for 31 such operators (Appendix B.2). It maps each operator step into an executable Python program with its input being the answers of the previous steps that it refers to.

Figure 4 displays the decomposition of our running example and its corresponding program operators. The answers to `qa_model` steps are annotated by workers, while the executor computes the answers of steps 4–6 based on their operator types and input (e.g., the group operator in step #4 is given the answers of steps 2–3 as its input).

3 Data Collection

This section describes the crowdsourcing process used to construct MoNaCo. As discussed in §2, we divided data collection into several parts: question writing and decomposition (§3.1), followed by the annotation of intermediate answers and evidence (§3.2). In §3.3 we describe our validation protocol to ensure the quality of our data. Lastly, estimating the potential human ‘‘ceiling performance’’ on MoNaCo is discussed in §A.1.

3.1 Question Writing and Decomposition

As described in §2.1, we ‘‘prompt’’ workers to write challenging information-seeking questions that would interest different target personas. We employed 24 Amazon Mechanical Turk crowd-workers who were paid \$0.9 per question. Overall, we collected 1,851 questions which, following decomposition, answer annotation, and validation, resulted in 1,315 fully annotated examples.

The second step in our annotation pipeline was question decomposition (§2.2). Given a question, workers were asked to write a series of intermediate steps using the QDMR syntax, (i.e., the step templates in Appendix B.2). For non-operator steps, workers were instructed to label whether the intermediate question is expected to have a single answer or a list of answers. This label was later used to set the price of intermediate answer annotation (§3.2). We employed four expert crowd-workers, who were already familiar with QDMR syntax, and paid them \$0.5 per decomposition.

3.2 Intermediate Answers and Evidence

Using the annotated decompositions and our execution engine, we were able to automatically derive intermediate questions which were then issued to workers (§2.2). Workers were asked to provide the answer and relevant evidence, using the English version of Wikipedia. The original (complex) question was also provided as context to help resolve any potential ambiguity. In addition, workers were asked to provide supporting evidence in the form of: (1) the relevant Wikipedia page, (2) the section containing the answer, and (3) the evidence sentence, or column if the evidence is a table. If a worker determined a question is unanswerable given the information in Wikipedia, they were asked to write a short justification. Our method ensured that all answers in MoNaCo are fully attributed to Wikipedia (Bohnet et al., 2022) with evidence at varying levels of granularity (page, section, sentence).

Workers were paid \$0.5 for answering list questions and \$0.2 for factoid questions (single answer). We collected 90K intermediate questions, answers and evidence, using 10 crowd workers.

3.3 Data Validation

We placed a strong emphasis on ensuring the quality and validity of MoNaCo data. To achieve this, we employed a range of validation methods, some general, such as worker qualification and feedback mechanisms, and others tailored to the specific requirements of each annotation task. For all of our tasks we enlisted Mechanical Turk workers from English-speaking countries, with over 5,000 HITs and an approval rating higher than 95%. For each of the annotation tasks we held preliminary qualifications and selected the top performing workers.

Manual Review and Feedback Similar to prior benchmarks, we conducted periodic reviews for quality control (Yu et al., 2018; Zhu et al., 2024) and provided ongoing feedback to workers—a practice that has been empirically shown to improve annotation quality (Scholman et al., 2022). All questions and decompositions (§3.1) were reviewed by one of the paper authors. We ensured that questions were grammatically correct, sufficiently complex and likely to be answerable using information from Wikipedia. In the question decomposition task, 14% of the annotations were corrected by the authors, primarily due to: (a) QDMR syntax errors and (b) decompositions containing questions that were unanswerable based on Wikipedia. For intermediate answer annotation (§3.2), we provided ongoing feedback to maintain annotation quality, reviewing 2,079 intermediate QA tasks in total. Lastly, the authors manually validated the final answers to all of the complex questions, resulting in 1,315 validated examples.

Factoid QA Validation To reduce annotation costs, and motivated by the near human performance of LLMs on factoid QA tasks (OpenAI, 2024; Wang et al., 2024), we chose to validate single answer questions using a hybrid approach: combining workers and a frontier LLM.

To validate worker answers we: (1) prompt an LLM for factoid QA, and (2) prompt a second LLM to judge the worker-LLM answer agreement. Using an LLM as a judge has become a standard approach (Phan et al., 2025; Wei et al., 2025) as lexical matching is often insufficient for measuring answer equivalence (Kamalloo et al., 2023). If the LLM judged the two answers to be distinct, we re-issued the question to a second worker for annotation. The process was repeated until two answers were judged to be in-agreement. We used GPT-4o for both factoid QA and for judging answer agreement (prompts in Figures 15–16, Appendix § B.4).

List QA Validation List questions are responsible for deriving multiple intermediate questions (§2.2) and unlike factoid questions, answering them remains challenging for LLMs (Amouyal et al., 2023; Malaviya et al., 2023). We therefore validate list questions manually, having two crowd-workers annotate each question. Worker answers were then compared based on their: (1) answer overlap and (2) length difference. The *answer overlap* between lists l_i, l_j is $\max\{contained(i,j), contained(j,i)\}$, where $contained(i,j) := |l_i \cap l_j|/\max\{|l_i|, |l_j|\}$. We consider item $k \in l_i$ to be in $l_i \cap l_j$ if there exists a $k' \in l_j$ such that the tokens of k are all in the set of tokens of k'. The *answer length difference* of lists l_i, l_j is normalized by $\max\{|l_i|, |l_j|\}$.

Answers with an overlap score higher than 0.77 and normalized length difference below 0.25 were considered to be correct, with the overlapping answers being treated as the gold answer list. If these criteria were not met, the question was re-issued to another worker for annotation, until two of the annotated lists would pass both criteria.

Unanswerable Questions Workers who were unable to answer their question, given the information in Wikipedia, labeled the question unanswerable along with a short justification. Overall, 5,363 intermediate questions were labeled as unanswerable. Of these, 2,398 questions were re-issued to a second worker, since their justifications were lacking or missing. Following re-annotation, the number of unanswerable intermediate questions was reduced to 4,083 (4.5% of the data).

4 The MoNaCo Dataset

This section provides an in-depth analysis of the MoNaCo data. In §4.1 we break down questions based on their type, decomposition, intermediate documents and answers—statistics that highlight the diverse nature of our data. Next, we empirically assess whether we have achieved our goal of collecting questions that are *more complex* than existing benchmarks (§4.2) while also *more natural* than other complex QA benchmarks (§4.3).

Complex questions:	1,315
- Avg. # question tokens	14.5
- Avg. # decomp. tokens	37.6
- Avg. # reasoning steps	5.1
- Aggregate operators %	45.3%
Intermediate questions:	90,773
- List I Boolean I other	8,549 I 40,125 I 42,099
- Evidence Wiki pages	36,194
- Sent. I Table I List %	29.5% I 67.8% I 2.7%
- # Wiki pages per Q	43.3 (avg) I 12 (median)
- # int. questions per Q	66.5 (avg) I 23 (median)
- # int. answers per Q	152.5 (avg) I 53 (median)

Table 2: MoNaCo data statistics.

4.1 Data Statistics

Table 2 summarizes the key statistics of our benchmark. MoNaCo questions are shorter than those in popular complex QA benchmarks, with 14.5 words on average compared to 17.5 in HotpotQA, 15.7 in Musique, and 17.3 in FanOutQA. While questions in MoNaCo are shorter, they entail far more intermediate steps than all these benchmarks (§4.2). This highlights how more natural questions can often entail complex reasoning using relatively concise language.

The 1,315 complex questions in MoNaCo have 90K intermediate questions, including 8,549 *list questions*, each list having 16.2 answers on average and a median of 5 answers. For comparison, the list QA benchmarks QAMPARI and QUEST have 2,000 and 3,357 manually written questions, making MoNaCo the largest benchmark of human-written list questions. MoNaCo also contains 40,125 Boolean (yes/no) questions, much more than past benchmarks such as BoolQ (Clark et al., 2019) and StrategyQA (Geva et al., 2021) with 15,942 and 2,835, respectively.

The intermediate answers are supported by evidence from 36,194 distinct Wikipedia pages. As described in §3.2, the evidence is either a sentence (29.5%), a table (67.8%) or a list (2.7%). This underscores the multi-modal aspect of MoNaCo, as answering its questions requires reasoning on both paragraphs and tables (Talmor et al., 2021).

4.2 How Complex are MoNaCo Questions?

To measure the complexity of MoNaCo questions, we compare it to 8 popular QA benchmarks: HotpotQA (Yang et al., 2018), DROP (Dua et al., 2019), Nat. Questions (Kwiatkowski et al., 2019), Mintaka (Sen et al., 2022), Musique (Trivedi et al., 2022), QUEST (Malaviya et al., 2023), and QAMPARI (Amouyal et al., 2023). Our analysis is presented in Table 3.

Benchmark	#Pages	#Steps	%Agg.	%Arith.	#Temp.	Divers.
DROP	1.0	2.8	12.5	**35.0**	39	2.89
FanOutQA	7.0	3.2	6.0	2.0	40	2.59
HotpotQA	2.0	2.2	2.5	1.0	36	2.39
Mintaka	1.8	2.4	12.0	2.0	33	2.71
Musique	2.6	2.6	2.5	2.5	30	1.93
Nat. Qs	1.0	1.1	2.5	1.0	13	0.56
QAMPARI	13.0	1.6	0.0	0.0	11	1.33
QUEST	10.5	2.9	0.5	0.0	22	2.25
MoNaCo	**43.3**	**5.0**	**38.0**	17.5	**119**	**4.48**

Table 3: Question complexity across benchmarks.

We first compared the average number of Wikipedia pages per question (#Pages), with MoNaCo averaging 43.3 evidence pages—more than thrice of that of the second highest benchmark. Next, we analyzed properties pertaining to the reasoning skills that these questions entail. To assess reasoning-related properties we represented questions using their question decomposition. For MoNaCo, questions were already annotated with decompositions, while for the remaining benchmarks we generated decompositions using a few-shot prompted GPT-4o.

We sampled 200 questions from each benchmark's test set, generated their decompositions and measured the following:

- Reasoning steps (#Steps): We list the average number of decomposition steps for the questions in each benchmark.
- Aggregation skills (%Agg.): The percentage of questions that require an aggregation operation (min, max, sum, average, median, group).
- Arithmetic skills (%Arith.): The percentage of questions entailing an arithmetic operation (addition, subtraction, division, multiplication).
- Reasoning templates (#Temp.): We define the sequence of decomposition step operators as the question's reasoning template, e.g. the template of the decomposition in Figure 4 is: *qa_model**3;*group_by*;*a_sorted_by_b*;*top_n*. A higher number of reasoning templates indicates that the benchmark is more challenging.
- Reasoning diversity (Divers.): We measure the diversity of questions in terms of their reasoning templates, computing Shannon's

diversity index (Shannon, 1948) over the unique templates from each benchmark.

The results in Table 3 demonstrate that MoNaCo questions are more reasoning-intensive than past benchmarks. The average number of reasoning steps is 5.1 compared to 3.2 for the second highest benchmark. Over 38% and 17% of MoNaCo questions require aggregate and arithmetic operations respectively, while its diversity is significantly higher than the other benchmarks.

4.3 How Natural are MoNaCo Questions?

A main goal in building MoNaCo was to ensure that its questions were *natural* and reflected the information-seeking goals of real-world users (de Vries et al., 2020; Bowman and Dahl, 2021). We followed Sen et al. (2022) in trying to measure how natural are MoNaCo questions compared to existing benchmarks: four containing natural questions (Nat. Questions, DROP, Mintaka, and FanOutQA) and four with questions that were partially machine generated (HotpotQA, Musique, QAMPARI, and QUEST).

Rather than having crowd-workers assign questions an underspecified (and rather ambiguous) ''naturalness'' score, we defined two criteria:

- Labeling *''who is more likely to have written the question?''* out of: (a) an expert user, (b) a regular user, (c) a machine rule-based script.
- A score between 1 and 5 indicating *''how likely does the person asking the question already know its answer?''*, 5 being extremely likely.

The first criterion is fairly straightforward, while the motivation behind the second is that, in real-world settings, people posing *information-seeking* questions are unlikely to be aware of the correct answer in advance. However, many questions in existing benchmarks give the impression that the question writer is already aware of the answer: *''What was triggered by a British Conservative Party politician?''*;*''Who is the sibling of the producer of Embedded in Baghdad?''* (Yang et al., 2018; Trivedi et al., 2022).

To score questions, we conducted a user study with 18 graduate students from the fields of natural language processing and data management. None of the paper authors took part in the user study. We randomly sampled 900 questions, with 100 questions from MoNaCo and each of the 8 aforementioned benchmarks test sets. Each participant was asked to score 50 random questions according to the guidelines in Appendix B.3. While participants were presented questions from different benchmarks they were unaware of their origin.

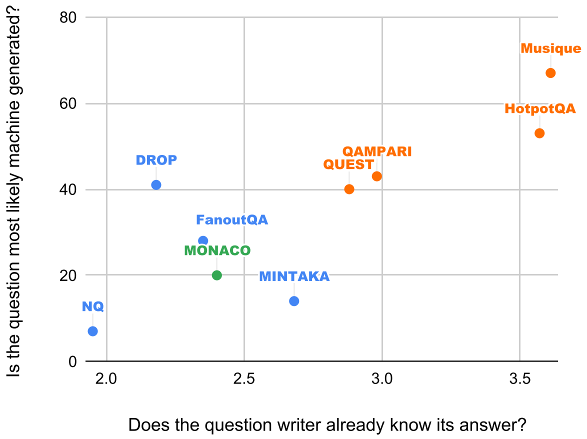

Figure 5: Measuring how natural are questions from different QA benchmarks. Benchmarks with natural questions are in blue while artificial ones are in orange.

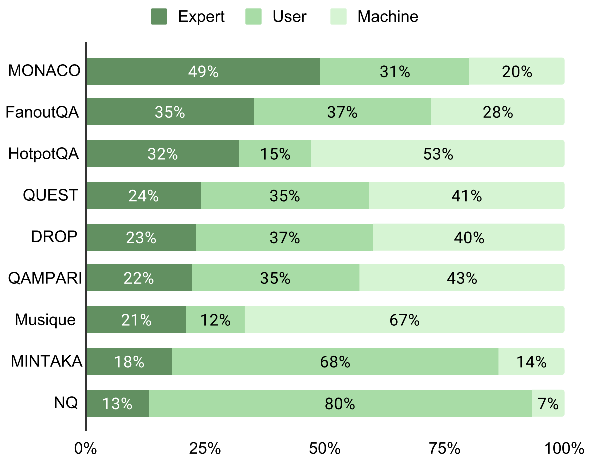

Figure 6: Breaking down benchmark questions based on their most likely writer: expert, user, or machine.

Figures 1, 5, and 6 present the user study results. The ''naturalness'' score in Figure 1 is a weighted average of both our measures, normalized between 0 and 100. In Figure 5 we plot each benchmark based on the percentage of questions labeled as machine-written (y-axis) and how likely was the question writer already aware of the answer (x-axis). The results demonstrate how questions from the natural benchmarks (in blue) tend to score lower on both our measures, in stark contrast to the more artificial benchmarks (in orange). MoNaCo questions (in green) appear much closer to the natural QA benchmarks than to the artificial ones. In Figure 6 we display the results of the most likely question writer. Somewhat unsurprising is that the top scoring Natural Questions (NQ) benchmark has only 7% of its questions labeled as being machine-written compared to 20% in MoNaCo. However, MoNaCo has 49% of its questions labeled as being expert-written compared to 13% in NQ.

Overall, this study highlights MoNaCo as more aligned with natural QA benchmarks while being more likely to have been expert-written—results that are fully in-line with our goals.

5 Experiments

We evaluated 15 frontier LLMs on MoNaCo and analyzed their performance across various types of questions and prompting methods. First, we describe our evaluation setup (§5.1) and present LLMs' closed-book results (§5.2). In §5.3 we analyze model performance based on the amount of intermediate steps and documents required to solve each question. Last, we examine LLM performance in a retrieval-augmented setting (§5.4).

5.1 Evaluation Setup

We evaluated 15 LLMs, including 8 models designed specifically for multi-step reasoning (OpenAI, 2024, 2025b; DeepSeek AI, 2025; Gemini Team, 2025; Anthropic, 2025). All models were prompted using the same instructions, provided in Figure 11 (Appendix B.4).

As string-based measures are often brittle (Kamalloo et al., 2023) we use GPT-4.1[2] as a judge to validate models predictions against the gold answers. We modified the evaluation prompt used by Phan et al. (2025) and Wei et al. (2025) to enable us to compute precision and recall automatically and in the case of numerical answers, to compute a normalized similarity score (Figure 12). For questions where the answer is a list of items, the LLM judge generates the total number of answers predicted by the model, followed by the list of predicted answers that also appear in the gold answers list. Both results enable us to compute precision and recall automatically based on the total number of predicted answers and the number of correctly predicted ones.

[2] `gpt-4.1-2025-04-14`.

Model	Precision	Recall	F1
GPT-5 (2025-08)	66.38	58.98	60.11
o3 (2025-05)	**68.10**	**59.54**	**61.18**
o4-mini (2025-04)	62.50	53.01	54.92
o3-mini (2025-04)	59.29	46.19	48.75
Claude 4-Opus (2025-07)	62.28	53.47	55.03
Gemini 2.5-Pro (2025-07)	65.02	58.14	59.11
Gemini 2.5-Flash (2025-07)	58.10	50.60	52.01
DeepSeek-R1	62.52	51.50	53.82
GPT-4o (2025-03)	57.37	46.98	48.98
+ few shot chains	63.33	52.88	55.05
GPT-4 Turbo (2024-05)	49.61	40.95	42.57
+ few shot chains	56.26	46.81	48.58
DeepSeek-V3	57.45	47.55	49.47
+ few shot chains	62.31	53.37	55.04
Llama 3.1-405B	55.03	46.28	47.67
+ few shot chains	55.97	51.20	51.39
Llama 3-70B	51.40	43.47	44.76
+ few shot chains	55.15	45.12	47.00
Qwen 2.5-72B	50.49	41.08	42.85
+ few shot chains	53.84	45.48	47.05
Qwen 2-72B	51.03	40.94	42.64
+ few shot chains	50.80	42.89	43.92

Table 4: Performance on MoNaCo of reasoning (top) and non-reasoning LLMs (bottom). Dashed lines separate closed-weights models from open ones. We underline the top performing model of each category.

We note that the predictions of all of the evaluated models, as well as the judgment scores generated by the LLM judge, are publicly released along with the MoNaCo data. For Claude 4-Opus we enabled its ''extended thinking'' with a ''budget tokens'' value of 6,000. For Gemini 2.5-Pro and Gemini 2.5-Flash we asked to ''include thoughts'' using the default ''dynamic thinking'', where the model decides how much to think. GPT-5 was evaluated using the default parameters for its ''reasoning effort'' (medium).

5.2 Language Models Performance

In this setting we evaluated the parametric knowledge of LLMs together with their ability to reason and aggregate over hundreds of facts. The results in Table 4 show that all LLMs fall significantly short of achieving a perfect score on MoNaCo. Even the top performing model, OpenAI's o3, reaches an F1 score of 61.2%, leaving substantial headroom for improvement. We observe that reasoning LLMs generally outperform strong non-reasoning models such as GPT-4o and Llama 3.1-405B. Notably, both DeepSeek-R1 and DeepSeek-V3 models performed quite well and are among the strongest models evaluated.

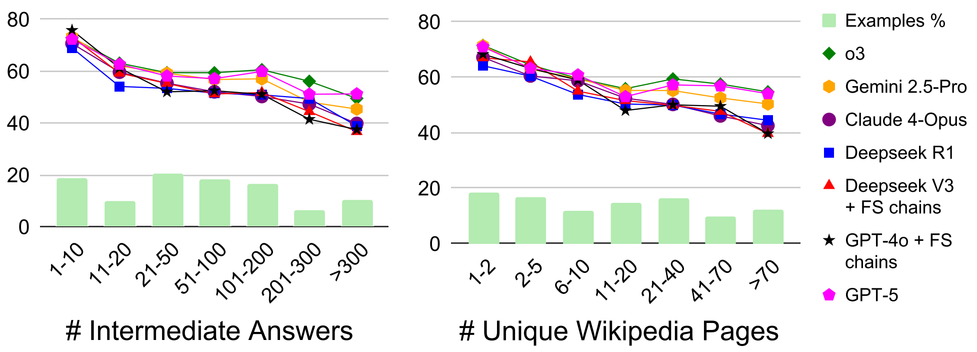

Figure 7: Complex QA performance (F1) as a function of the number intermediate answers / evidence pages.

Chain-of-Thought Prompting using a chain-of-thought (CoT) has been shown to improve performance on reasoning-heavy tasks (Wei et al., 2022; Kojima et al., 2022). We therefore sought to test whether CoT-prompting could further improve LLMs' performance on MoNaCo. We evaluated non-reasoning LLMs[3] using three CoT prompt variants: (1) a zero-shot prompt with the ''think step-by-step'' instructions; (2) a few-shot prompt with questions, answers and short explanations (Figure 17); (3) a few-shot prompt with full reasoning chains, including all of the intermediate questions, answers and reasoning steps (Figure 18). For both few-shot prompts (2), (3) we included the same 21 examples which cover all of the operator types in MoNaCo (Table 7).

The LLMs prompted using variants (1) and (2) did not perform any better than our original non-CoT prompt. The key improvement we observed was in using prompt (3), where the few-shot examples contained entire reasoning chains (Figure 18, Appendix B.4), explicitly prompting the LLM to generate all intermediate reasoning steps before its answer. Ultimately, the results in Table 4 show that all non-reasoning LLMs improve when prompted with few-shot reasoning chains.

5.3 Performance Breakdown and Analysis

Breakdown by Question Properties We break down model performance based on the amount of intermediate steps and factual information entailed by MoNaCo questions. Namely, we group questions based on the number of: (1) intermediate answers, and (2) distinct Wikipedia pages. Figure 7 presents the performance breakdown results, highlighting how all models struggle as the number of intermediate answers and evidence increases. However, o3 and GPT-5 performance is an exception: while it is on par with the other LLMs when the number of pages or intermediate answers is small, the gap increases significantly as the number of Wiki pages is in the dozens (>20).

[3] Following OpenAI and DeepSeek guidelines, we did not experiment with CoT-prompting the reasoning LLMs.

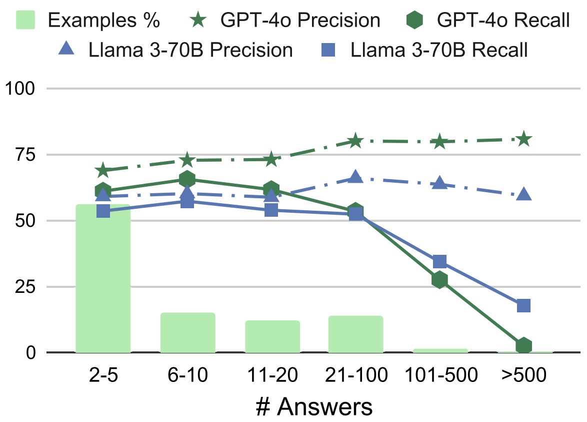

Figure 8: List QA prompted LLMs performance on the intermediate list questions in MoNaCo. We provide the average precision and recall scores. The full results are provided in Appendix A.2.

List Questions Questions in MoNaCo generally entail solving intermediate steps that have a list of answers. List questions have been shown to be challenging for past LLMs (Amouyal et al., 2023; Malaviya et al., 2023) and we revisit this challenge by evaluating GPT-4o and Llama 3-70B on all 8,549 intermediate list question in MoNaCo. Both models were prompted to generate an exhaustive list of answers using the few-shot prompt in Figure 14 (Appendix B.4). Due to cost limitations, we evaluate list QA precision and recall using string-based measures: matching each predicted answer to its most similar item in the gold answers based on their tokens F1 score. The results in Figure 8 demonstrate that list QA remains

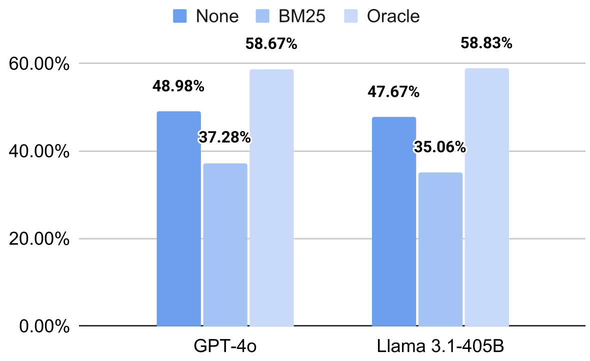

Figure 9: Retrieval-augmented LLM performance on MoNaCo (F1 score), with documents retrieved by BM25 or given the gold evidence documents (Oracle). The full results are provided in Appendix A.3.

challenging for frontier LLMs. While GPT-4o precision is high (75–80%) its recall sharply decreases with the number of expected answers.

5.4 Retrieval Augmented LLMs

Oracle Retrieval Unlike the experiments in §5.2, where models rely solely on their parametric knowledge, we also evaluated the complex reasoning skills of models over long contexts, when all the relevant evidence to MoNaCo questions is provided as input. This is an *Oracle retrieval* setting, where the LLM is provided all of the gold evidence documents in its prompt, effectively evaluating its reasoning skills in isolation from the task of knowledge retrieval. We evaluate GPT-4o and Llama 3.1-405B, and use the prompt in Figure 13 (Appendix B.4) to indicate that all input documents are indeed relevant. The Oracle results in Figure 9 show that both LLMs experience a 10 point improvement compared to the closed-book setting (None). Nevertheless, the fact that they reach only 58.7% indicates that even with all relevant knowledge, the complex reasoning in MoNaCo questions remains a challenge.

End-to-End RAG Retrieval-augmented generation (RAG) (Lewis et al., 2020) is a highly realistic setting where the LLM is provided with retrieved documents. We constructed a new retrieval index for Wikipedia, with 80M documents, since unlike the paragraph-specific indices used in prior work (Kwiatkowski et al., 2019), documents in MoNaCo can be either paragraphs, tables or lists. Our retriever is BM25 (Robertson et al., 2009) and we provide its top-20 documents as input to the LLM prompt (Figure 13, Appendix B.4). Ideally, RAG should improve over the closed-book setting, with LLMs utilizing information outside of their parametric knowledge while ignoring any irrelevant documents. However, both RAG models experience a sharp drop in performance compared to the original closed-book setting (Figure 9). While BM25 documents may only be partially relevant, our RAG prompt explicitly instructs to ignore the retrieved documents in such cases. While this lack of ‘‘retrieval robustness’’ (Yoran et al., 2024b) has been observed in weaker models, the limitation appears to persist even in current frontier LLMs.

6 Related Work

Information-seeking question answering is a longstanding approach to evaluate the reasoning skills of LLMs (Rodriguez and Boyd-Graber, 2021). Natural QA benchmarks consist almost entirely of ‘‘simple’’ questions, where the answers can be found within a single span of text (Bajaj et al., 2016; Abujabal et al., 2019; Jin et al., 2019; Krithara et al., 2023; Wei et al., 2024). As the reasoning skills of models improved, researchers have sought to evaluate LLMs on questions that require multi-hop reasoning skills (Yang et al., 2018; Geva et al., 2021; Trivedi et al., 2022; Press et al., 2023; Wei et al., 2025), as well as questions whose answer is a set of items which is spread across many documents (Amouyal et al., 2023; Malaviya et al., 2023; Zhong et al., 2023; Katz et al., 2023; Rassin et al., 2024).

Recent works introduced benchmarks featuring more naturally elicited questions that involve multiple documents (Sen et al., 2022; Zhu et al., 2024; Malaviya et al., 2024; Krishna et al., 2024; Yang et al., 2024; Phan et al., 2025). However, solving the questions in each of these benchmarks involves no more than a handful of webpages. By contrast, MoNaCo questions are significantly more time-consuming, as the evidence to answer each question typically involves dozens of pages.

More recently, Wei et al. (2025) introduced the BrowseComp benchmark for retrieving hard-to-find information. MoNaCo differs from this work in two main aspects. First, BrowseComp questions target niche topics and are phrased more as trivia questions, where the question writer already knows the answer and is trying to challenge someone else. In contrast, MoNaCo questions aim to convey the info-seeking goals of real-world users (§4.3). Second, the answers of BrowseComp questions are easy to verify, as their supporting evidence involves only a few webpages. MoNaCo questions are much broader, with answers that require to synthesize many pieces of evidence that are spread across dozens of Wikipedia pages.

WideSearch, a contemporary work by Wong et al. (2025), is a benchmark of real-world questions that evaluate LLM agents' ability to collect broad information. WideSearch contains 100 English questions, compared to 1,350 in MoNaCo.

Another key aspect in QA research has been benchmarking retrieval-augmented LLMs (Lewis et al., 2020; Ram et al., 2023). A great number of retrieval benchmarks focus on information-seeking tasks that are either domain-general (Voorhees and Tice, 2000; Thakur et al., 2021; Yang et al., 2024), domain-specific (Dasigi et al., 2021; Asai et al., 2024) or involve implicit reasoning skills (Geva et al., 2021; Hongjin et al., 2025). Nevertheless, the questions in all of these benchmarks rarely require to retrieve more than five evidence documents. The sheer breadth of evidence in MoNaCo position it as an advanced and more rigorous benchmark than many of the existing tasks for retrieval-augmented generation (Hsieh et al., 2024; Krishna et al., 2024) and factual attribution (Bohnet et al., 2022; Gao et al., 2023; Jacovi et al., 2025).

7 Conclusion

Questions that require extensive research across multiple webpages are incredibly important, yet they are not represented in current LLM benchmarks. To this end we introduce MoNaCo, a benchmark of human-written, realistic and time-consuming questions. It contains 1,315 questions, manually annotated with gold standard reasoning chains that include 90K intermediate questions, answers and supporting evidence documents. The modest performance of frontier LLMs on MoNaCo suggests that reasoning over dozens of documents remains an open challenge. As such, MoNaCo serves as a unique testbed for evaluating LLMs-powered systems on much broader tasks, which span across lots of documents and demand extensive factual knowledge, information retrieval and reasoning skills.

8 Limitations

Time-Dependent Questions A key factor in the evaluation of information-seeking questions is the potential of certain answers to change over time (Zhang and Choi, 2021; Vu et al., 2024). In our experiments, we did not measure the effect that temporal-dependence may have on the accuracy of future models evaluated on MoNaCo. Nevertheless, we fully provide users with the necessary resources to account for time-dependent answers.[4] Each question in MoNaCo was manually labeled by the paper authors as to whether its answer is expected to: (a) remain as is; (b) change every few years; (c) change on a yearly basis. Overall, 49.8% of the questions are time-independent; 34.1% have answers that change every few years; and 16.1% of have answers that are expected to change each year. Furthermore, we release the full timestamp of each answer in MoNaCo (i.e. the answer annotation date), enabling users to determine the relevant time-frame for every question in our data. This data should facilitate future usage of MoNaCo and enables researchers to evaluate models' performance at particular points in time.

Deep Research Systems In our experiments, we did not evaluate any LLM-powered ''Deep Research'' systems (OpenAI, 2025a; Huang et al., 2025) nor did we evaluate multi-step RAG systems that iteratively retrieve, decompose, and answer complex questions—interleaving retrieval and QA (Asai et al., 2023; Trivedi et al., 2023; Yoran et al., 2023). Since answering MoNaCo questions requires to retrieve and synthesize information from dozens of different webpages, it can serve as a useful testbed for evaluating the capabilities of such Deep Research systems.

LLM as a Judge As discussed in §5.1, we used GPT-4.1 to compare the answers generated by models to the gold answers and to determine whether or not the two are equivalent. While this is a commonly used practice (Phan et al., 2025; Wei et al., 2025) the judge model may still make mistakes, predicting wrong answers to be correct and vice-versa. Nevertheless, we view this as a reasonable compromise and, as LLMs continue to improve, swapping GPT-4.1 for a ''stronger'' judge model should further improve future evaluation.

[4] `https://huggingface.co/datasets/allenai/MoNaCo_Benchmark`.

Acknowledgments

We would like to thank Ron Yachini of AI2 Israel for his help and support. We would also like to thank Ori Yoran, Chaitanya Malaviya, Ofir Press, Roee Aharoni, Avi Caciularu, Allen Chang, Peter Clark, Jonathan Berant, and Daniel Deutch for their feedback and insightful comments. Lastly, we wish to express our gratitude to our action editor and reviewers for their time and effort. This work was done while the first author was an intern at the Allen Institute for AI. In addition, this project was partially funded by ONR contract N00014-23-1-2364. The last author was funded by The Israel Science Foundation grant 670/23, for which we are grateful.

References

Abdalghani Abujabal, Rishiraj Saha Roy, Mohamed Yahya, and Gerhard Weikum. 2019. Comqa: A community-sourced dataset for complex factoid question answering with paraphrase clusters. In *North American Association for Computational Linguistics (NAACL)*.

Samuel Amouyal, Tomer Wolfson, Ohad Rubin, Ori Yoran, Jonathan Herzig, and Jonathan Berant. 2023. QAMPARI: A benchmark for open-domain questions with many answers. In *Proceedings of the Third Workshop on Natural Language Generation, Evaluation, and Metrics (GEM)*, pages 97–110, Singapore. Association for Computational Linguistics.

Anthropic. 2025. System card: Claude opus 4 & claude sonnet 4.

Akari Asai, Jacqueline He, Rulin Shao, Weijia Shi, Amanpreet Singh, Joseph Chee Chang, Kyle Lo, Luca Soldaini, Sergey Feldman, Tian, D'arcy Mike, David Wadden, Matt Latzke, Minyang Tian, Pan Ji, Shengyan Liu, Hao Tong, Bohao Wu, Yanyu Xiong, Luke Zettlemoyer, Dan Weld, Graham Neubig, Doug Downey, Wen-tau Yih, Pang Wei Koh, and Hannaneh Hajishirzi. 2024. OpenScholar: Synthesizing scientific literature with retrieval-augmented language models. *arXiv preprint arXiv:2411.14199*.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2023. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. *arXiv preprint arXiv:2310.11511*.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. 2016. MS MARCO: A human generated MAchine Reading COmprehension dataset. *ArXiv*, abs/1611.09268.

Bernd Bohnet, Vinh Q. Tran, Pat Verga, Roee Aharoni, Daniel Andor, Livio Baldini Soares, Jacob Eisenstein, Kuzman Ganchev, Jonathan Herzig, Kai Hui, Tom Kwiatkowski, Ji Ma, Jianmo Ni, Tal Schuster, William W. Cohen, Michael Collins, Dipanjan Das, Donald Metzler, Slav Petrov, and Kellie Webster. 2022. Attributed question answering: Evaluation and modeling for attributed large language models. *ArXiv*, abs/2212.08037.

Samuel R. Bowman and George Dahl. 2021. What will it take to fix benchmarking in natural language understanding? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4843–4855, Online. Association for Computational Linguistics. `https://doi.org/10.18653/v1/2021.naacl-main.385`

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

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*, pages 4599–4610, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.naacl-main.365

Harm de Vries, Dzmitry Bahdanau, and Christopher Manning. 2020. Towards ecologically valid research on language user interfaces.

DeepSeek AI. 2025. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv*, abs/2501.12948.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. 2023. Enabling large language models to generate text with citations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6465–6488, Singapore. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.emnlp-main.398

Google Gemini Team. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *ArXiv*, abs/2507.06261.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? A question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361. https://doi.org/10.1162/tacl_a_00370

Mor Geva, Tomer Wolfson, and Jonathan Berant. 2022. Break, perturb, build: Automatic perturbation of reasoning paths through question decomposition. *Transactions of the Association for Computational Linguistics*, 10:111–126. https://doi.org/10.1162/tacl_a_00450

Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, N. Chawla, Olaf Wiest, and Xiangliang Zhang. 2024. Large language model based multi-agents: A survey of progress and challenges. In *International Joint Conference on Artificial Intelligence*.

S. U. Hongjin, Howard Yen, Mengzhou Xia, Weijia Shi, Niklas Muennighoff, Han-yu Wang, Liu Haisu, Quan Shi, Zachary S. Siegel, Michael Tang, et al. 2025. Bright: A realistic and challenging benchmark for reasoning-intensive retrieval. In *The International Conference on Learning Representations*.

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

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

Alon Jacovi, Andrew Wang, Chris Alberti, Connie Tao, Jon Lipovetz, Kate Olszewska, Lukas Haas, Michelle Liu, Nate Keating, Adam Bloniarz, Carl Saroufim, Corey Fry, Dror Marcus, Doron Kukliansky, Gaurav Singh Tomar, James Swirhun, Jinwei Xing, Lily Wang, Madhu Gurumurthy, Michael Aaron, Moran Ambar, Rachana Fellinger, Rui Wang, Zizhao Zhang, Sasha Goldshtein, and Dipanjan Das. 2025. The facts grounding leaderboard: Benchmarking llms' ability to ground responses to long-form input.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. 2019. PubMedQA: A dataset for biomedical research question answering. 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)*, pages 2567–2577, Hong Kong, China. Association for Computational Linguistics. https://doi.org/10.18653/v1/D19-1259

Ehsan Kamalloo, Nouha Dziri, Charles Clarke, and Davood Rafiei. 2023. Evaluating open-domain question answering in the era of large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5591–5606, Toronto, Canada. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.acl-long.307

Uri Katz, Matan Vetzler, Amir Cohen, and Yoav Goldberg. 2023. NERetrieve: Dataset for next generation named entity recognition and retrieval. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 3340–3354, Singapore. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.findings-emnlp.218

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *ICML 2022 Workshop on Knowledge Retrieval and Language Models*.

Satyapriya Krishna, Kalpesh Krishna, Anhad Mohananey, Steven Schwarcz, Adam Stambler, Shyam Upadhyay, and Manaal Faruqui. 2024. Fact, fetch, and reason: A unified evaluation of retrieval-augmented generation. https://doi.org/10.18653/v1/2025.naacl-long.243

Anastasia Krithara, Anastasios Nentidis, Konstantinos Bougiatiotis, and Georgios Paliouras. 2023. BioASQ-QA: A manually curated corpus for biomedical question answering. *Scientific Data*, 10(1):170. https://doi.org/10.1038/s41597-023-02068-4, PubMed: 36973320

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur 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. *Transactions of the Association for Computational Linguistics*, 7:452–466. https://doi.org/10.1162/tacl_a_00276

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Bangzheng Li, Ben Zhou, Fei Wang, Xingyu Fu, Dan Roth, and Muhao Chen. 2024. Deceptive semantic shortcuts on reasoning chains: How far can models go without hallucination? In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7675–7688, Mexico City, Mexico. Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.naacl-long.424

Bill Yuchen Lin, Yuntian Deng, Khyathi Chandu, Faeze Brahman, Abhilasha Ravichander, Valentina Pyatkin, Nouha Dziri, Ronan Le Bras, and Yejin Choi. 2024. Wildbench: Benchmarking llms with challenging tasks from real users in the wild.

Chaitanya Malaviya, Subin Lee, Sihao Chen, Elizabeth Sieber, Mark Yatskar, and Dan Roth. 2024. ExpertQA: Expert-curated questions and attributed answers. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3025–3045, Mexico City, Mexico. Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.naacl-long.167

Chaitanya Malaviya, Peter Shaw, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2023. QUEST: A retrieval dataset of entity-seeking queries with implicit set operations. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14032–14047, Toronto, Canada. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.acl-long.784

OpenAI. 2024. GPT-4 technical report.

OpenAI. 2024. OpenAI o1 system card.

OpenAI. 2025a. *Deep research system card*.

OpenAI. 2025b. Gpt-5 system card.

Long Phan, Alice Gatti, Ziwen Han, Nathaniel Li, Josephina Hu, Hugh Zhang, Chen Bo Calvin

Zhang, Mohamed Shaaban, John Ling, Sean Shi, Michael Choi, Anish Agrawal, Arnav Chopra, Adam Khoja, Ryan Kim, Richard Ren, Jason Hausenloy, Oliver Zhang, Mantas Maz, Summer Yue, Alexandr Wang, and Dan Hendrycks. 2025. Humanity's last exam.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah Smith, and Mike Lewis. 2023. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5687–5711, Singapore. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.findings-emnlp.378

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics. https://doi.org/10.18653/v1/D16-1264

Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11:1316–1331. https://doi.org/10.1162/tacl_a_00605

Royi Rassin, Yaron Fairstein, Oren Kalinsky, Guy Kushilevitz, Nachshon Cohen, Alexander Libov, and Yoav Goldberg. 2024. Evaluating D-MERIT of partial-annotation on information retrieval. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 2913–2932, Miami, Florida, USA. Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.emnlp-main.171

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389. https://doi.org/10.1561/1500000019

Pedro Rodriguez and Jordan Boyd-Graber. 2021. Evaluation paradigms in question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9630–9642, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.emnlp-main.758

Irina Saparina and Anton Osokin. 2021. SPARQLing database queries from intermediate question decompositions. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8984–8998, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.emnlp-main.708

Merel Scholman, Valentina Pyatkin, Frances Yung, Ido Dagan, Reut Tsarfaty, and Vera Demberg. 2022. Design choices in crowdsourcing discourse relation annotations: The effect of worker selection and training. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 2148–2156, Marseille, France. European Language Resources Association.

Priyanka Sen, Alham Fikri Aji, and Amir Saffari. 2022. Mintaka: A complex, natural, and multilingual dataset for end-to-end question answering. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1604–1619, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Claude Elwood Shannon. 1948. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423. https://doi.org/10.1002/j.1538-7305.1948.tb01338.x

Alon Talmor, Ori Yoran, Amnon Catav, Dan Lahav, Yizhong Wang, Akari Asai, Gabriel Ilharco, Hannaneh Hajishirzi, and Jonathan Berant. 2021. Multimodal{qa}: Complex question answering over text, tables and images. In *International Conference on Learning Representations*.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. MuSiQue: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554. https://doi.org/10.1162/tacl_a_00475

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10014–10037, Toronto, Canada. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.acl-long.557

Ellen M. Voorhees and Dawn M. Tice. 2000. Building a question answering test collection. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 200–207. https://doi.org/10.1145/345508.345577

Tu Vu, Mohit Iyyer, Xuezhi Wang, Noah Constant, Jerry Wei, Jason Wei, Chris Tar, Yun-Hsuan Sung, Denny Zhou, Quoc Le, and Thang Luong. 2024. FreshLLMs: Refreshing large language models with search engine augmentation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 13697–13720, Bangkok, Thailand. Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.findings-acl.813

Cunxiang Wang, Sirui Cheng, Qipeng Guo, Yuanhao Yue, Bowen Ding, Zhikun Xu, Yidong Wang, Xiangkun Hu, Zheng Zhang, and Yue Zhang. 2024. Evaluating open-qa evaluation. *Advances in Neural Information Processing Systems*, 36.

Jason Wei, Nguyen Karina, Hyung Won Chung, Yunxin Joy Jiao, Spencer Papay, Amelia Glaese, John Schulman, and William Fedus. 2024. Measuring short-form factuality in large language models.

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

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Tomer Wolfson, Daniel Deutch, and Jonathan Berant. 2022. Weakly supervised text-to-SQL parsing through question decomposition. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2528–2542, Seattle, United States. Association for Computational Linguistics. https://doi.org/10.18653/v1/2022.findings-naacl.193

Tomer Wolfson, Mor Geva, Ankit Gupta, Matt Gardner, Yoav Goldberg, Daniel Deutch, and Jonathan Berant. 2020. Break it down: A question understanding benchmark. *Transactions of the Association for Computational Linguistics*, 8:183–198. https://doi.org/10.1162/tacl_a_00309

Ryan Wong, Jiawei Wang, Junjie Zhao, Li Chen, Yan Gao, Long Zhang, Xuan Zhou, Zuo Wang, Kai Xiang, Ge Zhang, Wenhao Huang, Yang Wang, and Ke Wang. 2025. Widesearch: Benchmarking agentic broad info-seeking.

Xiao Yang, Kai Sun, Hao Xin, Yushi Sun, Nikita Bhalla, Xiangsen Chen, Sajal Choudhary, Rongze Gui, Ziran Jiang, Ziyu JIANG, Lingkun Kong, Brian Moran, Jiaqi Wang, Yifan Ethan Xu, An Yan, Chenyu Yang, Eting Yuan, Hanwen Zha, Nan Tang, Lei Chen, Nicolas Scheffer, Yue Liu, Nirav Shah, Rakesh Wanga, Anuj Kumar, Wen tau Yih, and Xin Luna Dong. 2024. CRAG - comprehensive RAG benchmark. In *The Thirty-eighth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William 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*, pages 2369–2380.

Brussels, Belgium. Association for Computational Linguistics. https://doi.org/10.18653/v1/D18-1259

Ori Yoran, Samuel Joseph Amouyal, Chaitanya Malaviya, Ben Bogin, Ofir Press, and Jonathan Berant. 2024a. Assistantbench: Can web agents solve realistic and time-consuming tasks? https://doi.org/10.18653/v1/2024.emnlp-main.505

Ori Yoran, Tomer Wolfson, Ben Bogin, Uri Katz, Daniel Deutch, and Jonathan Berant. 2023. Answering questions by meta-reasoning over multiple chains of thought. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5942–5966. Singapore. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.emnlp-main.364

Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. 2024b. Making retrieval-augmented language models robust to irrelevant context. In *The Twelfth International Conference on Learning Representations*.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. 2018. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921. Brussels, Belgium. Association for Computational Linguistics. https://doi.org/10.18653/v1/D18-1425

Michael Zhang and Eunsol Choi. 2021. SituatedQA: Incorporating extra-linguistic contexts into QA. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7371–7387. Online and Punta Cana, Dominican Republic. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.emnlp-main.586

Victor Zhong, Weijia Shi, Wen-tau Yih, and Luke Zettlemoyer. 2023. RoMQA: A benchmark for robust, multi-evidence, multi-answer question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 7055–7067. Singapore. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.findings-emnlp.470

Andrew Zhu, Alyssa Hwang, Liam Dugan, and Chris Callison-Burch. 2024. Fanoutqa: A multi-hop, multi-document question answering benchmark for large language models. https://doi.org/10.18653/v1/2024.acl-short.2

A Appendix: Supplementary Results

A.1 Benchmark Ceiling Performance

Several question answering benchmarks have included estimates of the human ceiling performance on their task (Rajpurkar et al., 2016; Yang et al., 2018; Dua et al., 2019; Geva et al., 2021). Such benchmarks typically contain questions which are not time-consuming and can be solved relatively fast by a single human annotator. By contrast, MoNaCo questions are time-consuming and require to combine information from 43 distinct webpages on average. This inherent complexity is what encouraged us to divide our benchmark's annotation process into multiple, much simpler steps, as described in §3.

Rather than estimating human performance directly, we sought to ensure that the collected answers to MoNaCo questions were indeed valid (§3.3). Question decompositions were manually reviewed; factoid question answers were human-annotated, then validated by GPT-4o, or a second crowd-worker if the first worker's answer contradicted that of the LLM. In the case of intermediate ‘‘list questions’’ there was a higher risk of ambiguity in the question. For example, in questions like *‘‘What are some vegan dishes in Chile?’’* or *‘‘Who are female poets of Andalusian Spain?’’*. However, the answers to such questions are unambiguous given our underlying information-source—the English version of Wikipedia. Furthermore, the validation process of list QA demanded a high answer overlap between two human annotators (>77%), which further indicates our list questions are unambiguous, given Wikipedia.

Lastly, the execution of operator steps such as aggregation or numerical comparison is performed automatically, leaving no room for potential calculation errors by humans.

A.2 List QA Performance

Table 5 provides the detailed performance breakdown of models over the intermediate list questions in MoNaCo. The results are displayed as a plot in Figure 8.

Model	# Ans.	# Ex.	Pre.	Rec.	F1
GPT-4o	2–5	4,812	68.9	61.1	62.3
	6–10	1,312	72.8	65.6	66.5
	11–20	1,064	73.1	61.7	64.5
	21–100	1,206	80.1	53.5	60.7
	101–500	137	79.8	27.6	37.6
	>500	18	80.8	2.5	4.8
	All:	8,549	76.0	61.8	64.5
Llama 3-70B	1–5	4,812	59.1	53.9	53.6
	6–10	1,312	60.2	59.8	57.3
	11–20	1,064	58.8	53.8	53.9
	21–100	1,206	66.0	47.6	52.4
	101–500	137	63.7	28.2	34.4
	>500	18	59.4	19.0	17.9
	All:	8,549	60.3	53.3	53.7

Table 5: List QA prompted LLMs performance on the intermediate list questions in MoNaCo. We provide the average precision, recall, and F1 scores.

A.3 Retrieval Performance

Table 6 provides the detailed results of the RAG experiments on MoNaCo (also plotted in Figure 9).

Model	Retriever	Pre.	Rec.	F1
GPT-4o	None	57.37	46.98	48.98
	BM25	48.21	34.94	37.28
	Oracle	67.28	56.08	58.67
Llama 3.1-405B	None	55.03	46.28	47.67
	BM25	44.18	33.32	35.06
	Oracle	66.68	56.57	58.83

Table 6: LLM performance on MoNaCo given gold evidence (Oracle) and retrieved documents (BM25) as input.

B Appendix: Additional Implementation Details

B.1 Eliciting Complex Natural Questions: Additional Details

In each question-writing task, workers were provided with: (1) a persona description, (2) helpful keywords, and (3) reference questions from different personas. We came up with 28 target personas, covering a wide range of domains: geography, demographics, politics, world leaders, economics, higher education, languages, history, nutrition, cuisine, sports, film, television, music, literature, and theater. Each persona was *described* in 2–4 sentences, as shown in Part 1 of Figure 2. To promote diversity, workers received seven *keywords* which were randomly selected from a larger persona-specific pool. To encourage the creation of challenging questions, we also included five *reference questions* (initially written by us and later replaced by worker-written ones) which were intentionally time-consuming, often involving dozens of documents. Importantly, we were careful to use reference questions from personas that were *unrelated* to the target persona, thereby discouraging workers from simply paraphrasing reference questions. To further promote creativity, we avoided restricting workers to a set of pre-defined question templates. Finally, in order to reduce any potential reasoning shortcuts in questions, we did not provide workers with any of the answers (or evidence) in advance.

Overall, this approach enabled us to collect questions that were judged as both realistic and significantly more time-consuming than those in existing benchmarks §4.3.

B.2 Decomposition Execution

Table 7 lists all query operators supported by our executor and their corresponding Python function signatures. This implementation extends the original QDMR representation in Wolfson et al. (2020) to 31 operators.

B.3 Natural Questions User Study

Figure 10 provides the full guidelines given to the participants of our user study that measures how natural are the questions in MoNaCo, compared to those in other information-seeking QA benchmarks. For the user study details see §4.3.

B.4 Prompts

This section describes the prompts used throughout the paper. Please note that some of the prompts have only part of their examples listed, to save space. For the full prompt descriptions please refer to our project's repository at: `https://tomerwolgithub.github.io/monaco/`.

Operator	Decomposition Template	Python Function Signature
Question	*A free-form question*	`qa_model(question: str) -> str | list[str]`
Yes/no filter	return #x where #y is {true / false}	`filter_boolean(entities: list[str], booleans: list[bool], required_value: bool = True) -> list[str]`
Comparison filter	return #x where #y {comparator} {val}	`filter_compare(entities: list[str], left_values: list[float], comparator: Literal[">", "<", ">=", "<=", "=="], right_value_or_values: Number | datetime | list[float] | list[datetime]) -> list[str]:`
Superlative filter	return #x where #y is {highest / lowest}	`filter_superlative(entities: list[str], values: list[float], superlative: Literal["max", "min"]) -> list[str]`
Sort	return #x sorted {by #y} {ascending / descending}	`a_sorted_by_b(items_a: list[str], items_b: list[Number | datetime], reverse: bool = False) -> list[str]`
Set intersection	return {#x / val} in both #y and #z	`items_in_both(list_of_items_a: list[str], list_of_items_b: list[str]) -> list[str]`
Set union	return #x, #y {, #z, ...}	`concatenate_items(*items: str | Number | datetime) -> list[str | Number | datetime]`
Set difference	return #x besides { #y / val}	`discard(items: list[str], items_to_discard: str | list[str]) -> list[str]`
Top-n	return the top {num} of #x	`top_n(items: list[str], n: int) -> list[str]`
Item at n'th position	return the {ordinal} of #x	`access_list_index(items: list[str], n: int) -> str`
Addition	return sum of #x and #y	`addition(number_or_list_1: Number | list[Number], number_or_list_2: Number | list[Number]) -> Number | list[Number]`
Subtraction	return difference of #x and #y	`difference(number_or_list_1: Number | list[Number], number_or_list_2: Number | list[Number]) -> Number | list[Number]`
Multiplication	return multiplication of #x and #y	`multiplication(number_or_list_1: Number | list[Number], number_or_list_2: Number | list[Number]) -> Number | list[Number]`
Division	return division of #x and #y	`division(number_or_list_1: Number | list[Number], number_or_list_2: Number | list[Number]) -> Number | list[Number]`
Percentage	return percentage of #x and #y	`percentage(number_or_list_1: Number | list[Number], number_or_list_2: Number | list[Number]) -> Number | list[Number]`
Count	return number of #x	`count(items: list[str]) -> int`
Average	return average of #x	`average(items: list[Number]) -> Number`
Median	return median of #x	`median(items: list[Number]) -> Number`
Max	return highest of #x	`max(items: list[Number]) -> Number`
Min	return lowest of #x	`min(items: list[Number]) -> Number`
Sum	return sum of #x	`sum(items: list[Number]) -> Number`
Group	return the {aggregate} of #y for each #x	`group_by(entities: list[str], aggregator: Literal["count", "sum", "average", "median", "min", "max"], values: list[Number | datetime]) -> dict[str, Number | datetime] | list[str, Number | datetime]`
==	return if #x is equal to {#y / val}	`equals(item_1: str | Number | datetime, item_2: str | Number | datetime) -> bool`
<	return if #x is less than {#y / val}	`greater_than(item_1: Number | datetime, item_2: Number | datetime) -> bool`
>	return if #x is higher than {#y / val}	`less_than(item_1: Number | datetime, item_2: Number | datetime) -> bool`
>=	return if #x is at most {#y / val}	`at_least(item_1: Number | datetime, item_2: Number | datetime) -> bool`
<=	return if #x is at least {#y / val}	`at_most(item_1: Number | datetime, item_2: Number | datetime) -> bool`
Boolean intersection	return if both #x and #y are {true / false}	`both_true(items_a: bool, items_b: bool) -> bool`
Arg max	return which is highest of #x, #y {, #z, ...}	`argmax(*items: Number) -> Number`
Arg min	return which is lowest of #x, #y {, #z, ...}	`argmin(*items: Number) -> Number`
Arg true	return which is true of #x, #y {, #z, ...}	`which_is_true(items_a: bool, items_b: bool) -> bool`

Table 7: All of the query operators supported by our decomposition executor.

Guidelines

You are asked to rank 50 open-domain questions. We wish to rank how much does a question reflect a *likely information-seeking goal* posed by a user.
We score questions based on their: **(a) information-seeking nature**, **(b) who is posing the question (writer)**.
See examples for these criteria below.

a. For each question, score how likely *does the person asking the question know its answer?*
5 - very likely that the person asking know the answer in advance
4 - somewhat likely
3 - not very likely
2 - unlikely
1 - very unlikely that the person is familiar with the answer

b. Fill in who is most likely to be posing the questions (question writer), out of the following:
* Expert: question written by an invidual interested in a particular field, seeking to learn more about it
* User: question written by a non-expert user, posing an information-seeking question that interests them
* Machine: question automatically generated by a rule-based script

(a) Information-seeking questions:

* An *information-seeking question* is one where the answer provides new information, previously unknown to the user (e.g. "What types of diets should I try if I am suffering from diabetes?").

* Contrastly, a *trivia question* is one whose writer is already familiar with its answer in advance (e.g. "Which movie actor, who was married to Angelina Jolie, also starred in the film Troy?"). As we are interested in questions posed to search engines, users are unlikely to be interested in questions they already know the answers to.

(b) Types of question writers:

Based on three potential question writers (expert/user/machine) we want to identify who is the *most likely* one that wrote the question. Consider the following examples:

1. "How many wars has Britain lost in the last 300 years?" /
"In European countries, are current left-wing political parties more likely to be headed by women than right-wing ones?" – These questions ask about a complex, yet realistic, information-seeking goal. It is likely that an expert users, interested in a particular field, wrote these questions.

2. "Which Thai salads can be considered vegetarian friendly?" /
"Who played junior on in the heat of the night?" – These questions were likely written by a regular user interested in a particular subject.

3. "David Dobkin's 2007 comedy uses what song by Hawaiian singer Israel Kamakawiwo'ole?" /
"Who is the spouse of a character contained in The Farmer Boy?" – while these questions convey an info-seeking goal, it is not very likely. Their phrasing is somewhat unnatural and they most likely have been generated by a machine.

Figure 10: The guidelines for our user study measuring how natural are the questions of complex QA benchmarks.

System prompt:
You are a helpful question answering assistant. Your task is to answer a complex question provided by the user. You may generate an explanation before providing the answer. The answer must be generated as a concise list of one or more entities, numbers or dates. You must always answer the question, even if your information is not up-to-date, please answer based on it.
Your response must use the following format:
Answers: {ANSWERS}
Where ANSWERS is a list of potential answers, separated by commas. You must end your response after the final answer. You must always answer the question, even if your information is not up-to-date, please answer based on it.

Figure 11: Complex QA system prompt, used to benchmark LLMs performance on MoNaCo.

System prompt:
Judge whether the following [response] to [question] is correct or not based on the precise and unambiguous [correct_answer] below.
[question]: {question}
[response]: '{response}'

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

extracted_final_answer: The final exact answer extracted from the [response]. Put the extracted answer as 'None' if there is no exact final answer to extract from the response.

[correct_answer]: {correct_answer}

final answer length: Provide the overall number of unique answers that appear in [response], not just the correct ones. Be sure to provide a number, not an estimate!

reasoning: Explain why the extracted_final_answer is correct or incorrect based on [correct_answer], focusing only on if there are meaningful differences between [correct_answer] and the extracted_final_answer. Do not comment on any background to the problem, do not attempt to solve the problem, do not argue for any answer different than [correct_answer], focus only on whether the answers match.

correct: Answer 'yes' if extracted_final_answer matches the [correct_answer] given above, or is within a small margin of error for numerical problems, a margin of 1 to 5.5 percentage points is acceptable. Answer 'no' otherwise, i.e. if there is any inconsistency, ambiguity, non-equivalency, or if the extracted answer is incorrect.

precision: Answer '1' if extracted_final_answer matches the [correct_answer] given above. Answer '0' otherwise, i.e. if there is any inconsistency, ambiguity, non-equivalency, or if the extracted answer is incorrect. In the case where [correct_answer] is a number or percentage, then answer with the following formula to compute the normalized similarity score:
[1 - (abs([correct_answer] - extracted_final_answer) / max(abs([correct_answer]), abs(extracted_final_answer)))]

final precision: Extract the precision score from above, just the final score (number).

overlapping answers: List all of the answers in [response] that also appear in [correct_answer]. You can consider an answer from [response] to match with an answer in [correct_answer] if it is equivalent or is within a small margin of error for numerical problems, a margin of 1 to 5.5 percentage points is acceptable. List all of the [response] answer appearing in [correct_answer] with each answer delimited by '###'. If the number of overlapping answers is zero, output 'NULL'.

Figure 12: The LLM-as-judge evaluation prompt.

Oracle retrieval:
*** Above are multiple excerpts of paragraphs and tables, each of document has a title, followed by the actual content. These documents contain highly relevant information which should help you answer the user question.

BM25 retrieval:
*** Above are provided with multiple excerpts of paragraphs and tables, each of document has a title, followed by the actual content. Some of these documents might contain helpful information to answering the question. In case that the information in the document is relevant, you may use it to solve the question. If a document is irrelevant feel free to ignore it when answering.

Figure 13: Retrieval-augmented QA instructions which follow the same system prompt from Figure 11.

System prompt:
You are a helpful question answering assistant. Your task is to answer a complex question provided by the user. You may generate an explanation before providing the answer, as a chain-of-thought reasoning. The answer must be generated as a concise list of one or more entities, numbers or dates. You must always answer the question, even if your information is not up-to-date, please answer based on it.
Your response must use the following format:
Let's think step by step: {EXPLANATION}
Answers: {ANSWERS}
Keep the explanation as concise as possible. ANSWERS is a list of potential answers, each answer in a separate line. You must end your response after the final answer. You must always answer the question, even if your information is not up-to-date, please answer based on it.

Prompt examples:

Question: What are all the species of bears that currently exist in the world?
Let's think step by step: To find the answer we will need to return all the different types of bears that are not gone extinct.
Answers: American Black Bear, Asian Black Bear, Brown Bear, Polar Bear, Sloth Bear, Spectacled Bear, Sun Bear, Giant Panda

Question: List all of the offices in Italy's current cabinet?
Let's think step by step: To answer this question, we need to return all the ministerial positions of Italy's incumbent government.
Answers: Prime Minister, Deputy Prime Minister, Minister of Foreign Affairs and International Cooperation, Minister of the Interior, Minister of Justice, Minister of Defence, Minister of Economy and Finance, Minister of Business and Made in Italy, Minister of Agriculture, Food Sovereignty and Forests, Minister for the Environment and Energy Security, Minister of Infrastructure and Transport, Minister of Labour and Social Policies, Minister of Education and Merit, Minister of University and Research, Minister of Culture, Minister of Health, Minister of Tourism, Minister for Relations with Parliament, Minister for Public Administration, Minister for Regional Affairs and Autonomies, Minister for Civil Protection and Maritime Policies, Minister for European Affairs, Southern Italy, Cohesion Policy and the NRRP, Minister for Sport and Youth, Minister for Family, Birth Rate and Equal Opportunities, Minister for Disabilities, Minister for Institutional Reforms and Regulatory Simplification, Secretary of the Council of Ministers
...

Figure 14: Excerpt of the list QA prompt, used to evaluate LLM performance on list questions.

System prompt:
You are a helpful question answering assistant. Your task is to answer a complex question provided by the user. You may generate an explanation before providing the answer, as a chain-of-thought reasoning. The answer must be generated as a concise list of one or more entities, numbers or dates. You must always answer the question, even if your information is not up-to-date, please answer based on it.
Your response must use the following format:
Let's think step by step: {EXPLANATION}
Answers: {ANSWERS}
Keep the explanation as concise as possible. ANSWERS is a list of potential answers, each answer in a separate line. You must end your response after the final answer. You must always answer the question, even if your information is not up-to-date, please answer based on it.

Prompt examples:

Question: Is Toosie Slide a musical collaboration?
Let's think step by step: "Toosie Slide" is not a musical collaboration. It is a solo song by the Canadian rapper Drake. Released on April 3, 2020, the song became popular on social media platforms, especially TikTok, due to its catchy hook and dance challenge. The dance associated with the song was created by the dancer Toosie, which is where the song gets its name. However, there are no other artists featured in the track.
Answers: no
...

Figure 15: Excerpt of the factoid QA prompt, used for factoid QA validation.

System prompt:
You are a helpful assistant, tasked with determining whether two answers are in agreement with each other. Given a question and two potential answers, determine whether both answers are the same. If both answers refer to the same entity (e.g. 'US' and 'United States of America') or person ('President Biden' and 'Joe Bide') In the case of numeric answers, we regard numbers that are very close to each other as being similar. Your response should be either 'yes' or 'no'.

Prompt examples:

Question: For how long did Fahd serve as king of Saudi Arabia?
Answer 1: 23 years
Answer 2: 13 June 1982 – 1 August 2005
Are these answers the same? no

Figure 16: Excerpt of the answer agreement prompt, used for factoid QA validation.

System prompt:
You are a helpful question answering assistant. Your task is to answer a complex question provided by the user. You may generate an explanation before providing the answer, as a chain-of-thought reasoning. The answer must be generated as a concise list of one or more entities, numbers or dates. You must always answer the question, even if your information is not up-to-date, please answer based on it. Your response must use the following format:
Let's think step by step: {EXPLANATION}
Answers: {ANSWERS}
Keep the explanation as concise as possible. ANSWERS is a list of potential answers, separated by commas. You must end your response after the final answer. You must always answer the question, even if your information is not up-to-date, please answer based on it.

Prompt examples:

Question: How many plays did Arthur Miller write between the ages of 20 and 40?
Let's think step by step:
to answer the question we need to find out what were the years in which Arthur Miller turned 20 and 40. Then, we need to review the plays written by Miller and return only the plays that were written between those years.
Answers: 15

Figure 17: Excerpt of the few-shot chain-of-thought prompt (CoT).

System prompt:
You are a helpful question answering assistant. Your task is to answer a complex question provided by the user. You may generate an explanation before providing the answer, as a chain-of-thought reasoning. The answer must be generated as a concise list of one or more entities, numbers or dates. You must always answer the question, even if your information is not up-to-date, please answer based on it. Your response must use the following format:
Let's think step by step: {EXPLANATION}
Answers: {ANSWERS}
Keep the explanation as concise as possible. ANSWERS is a list of potential answers, separated by commas. You must end your response after the final answer. You must always answer the question, even if your information is not up-to-date, please answer based on it.

Prompt examples:

Question: How many plays did Arthur Miller write between the ages of 20 and 40?
Let's think step by step:
1. Arthur Miller was born on October 17, 1915, so he turned 20 on October 17, 1935 and he turned 40 on October 17, 1955.
2. Between October 17, 1935 and October 17, 1955 Arthur Miller wrote the following plays: No Villain (1936), They Too Arise (1937), Honors at Dawn (1938), The Grass Still Grows (1938), The Great Disobedience (1938), Listen My Children (1939), The Golden Years (1940), The Half-Bridge (1943), The Man Who Had All the Luck (1944), All My Sons (1947), Death of a Salesman (1949), An Enemy of the People (1950), The Crucible (1953), A View from the Bridge (1955), A Memory of Two Mondays (1955).
3. Therefore Miller had written 15 plays between October 17, 1935 and October 17, 1955.
Answers: 15

Figure 18: Excerpt of the few-shot chain-of-thought prompt which includes intermediate answers (CoT+Ans).