FutureHouse

LANGUAGE AGENTS ACHIEVE SUPERHUMAN SYNTHESIS OF SCIENTIFIC KNOWLEDGE

Michael D. Skarlinski[1] **Sam Cox**[1,2] **Jon M. Laurent**[1]

James D. Braza[1] **Michaela Hinks**[1] **Michael J. Hammerling**[1]

Manvitha Ponnapati[1] **Samuel G. Rodriques**[1,3*] **Andrew D. White**[1,2*]

[1]FutureHouse Inc., San Francisco, CA
[2]University of Rochester, Rochester, NY
[3] Francis Crick Institute, London, UK
*These authors jointly supervise technical work at FutureHouse.
Correspondence to: {sam,andrew}@futurehouse.org

ABSTRACT

Language models are known to "hallucinate" incorrect information, and it is unclear if they are sufficiently accurate and reliable for use in scientific research. We developed a rigorous human-AI comparison methodology to evaluate language model agents on real-world literature search tasks covering information retrieval, summarization, and contradiction detection tasks. We show that PaperQA2, a frontier language model agent optimized for improved factuality, matches or exceeds subject matter expert performance on three realistic literature research tasks without any restrictions on humans (i.e., full access to internet, search tools, and time). PaperQA2 writes cited, Wikipedia-style summaries of scientific topics that are significantly more accurate than existing, human-written Wikipedia articles. We also introduce a hard benchmark for scientific literature research called LitQA2 that guided design of PaperQA2, leading to it exceeding human performance. Finally, we apply PaperQA2 to identify contradictions within the scientific literature, an important scientific task that is challenging for humans. PaperQA2 identifies 2.34 ± 1.99 (mean $\pm$ SD, $N = 93$ papers) contradictions per paper in a random subset of biology papers, of which 70% are validated by human experts. These results demonstrate that language model agents are now capable of exceeding domain experts across meaningful tasks on scientific literature.

1 Introduction

Large language models (LLMs) have the potential to assist scientists with retrieving, synthesizing, and summarizing the literature [1,2,3], but still have several limitations for use in research tasks. Firstly, factuality is essential in scientific research, and LLMs hallucinate [4], confidently stating information that is not grounded in any existing source or evidence. Secondly, science requires extreme attention to detail, and LLMs can overlook or misuse details when faced with challenging reasoning problems [5]. Finally, benchmarks for retrieval and reasoning across the scientific literature today are underdeveloped. They do not consider the entire literature, but instead are restricted to abstracts [6], retrieval on a fixed corpus [7], or simply provide the relevant paper directly [8]. These benchmarks are not suitable as performance proxies for real scientific research tasks, and, more importantly, often lack a direct comparison to human performance. Thus, it remains unclear whether language models and agents are suitable for use in scientific research.

We therefore set out to develop a rigorous comparison between the performance of AI systems and humans on three real-world tasks: a retrieval task involving searching the entire literature to answer questions; a summarization task involving producing a cited, Wikipedia-style articles on scientific topics; and a contradiction-detection task, involving extracting all claims from papers and checking them for contradictions against all of literature. This is, to our knowledge,

Figure 1: **A**. Schematic of PaperQA2's agentic toolset along with relevant action representations within each tool. **B.** PaperQA2 performance across question answering, cited article summarization, and contradiction detection. Error bars represent standard error.

the first robust procedure for evaluating a single AI system on multiple real-world literature search tasks. Using our newly developed evaluations, we explored multiple designs leading to a system we call PaperQA2 (**Figure 1A**), which exceeds the performance of PhD students and postdocs on the retrieval and summarization tasks. Applying PaperQA2 to the contradiction detection task enables us to identify contradictions in biology papers at scale (**Figure 1B**). For example, a statement that the ZNF804A rs1344706 allele positively affects brain structure in schizophrenia patients[9] was found to be contradicted by a later publication which found that rs1344706's effects on cortical thickness, surface area, and cortical volume in the brain aggravate the risk of schizophrenia[10].

2 Answering scientific questions

To evaluate AI systems on retrieval over the scientific literature, we first generated LitQA2,[11] a set of 248 multiple choice questions with answers that require retrieval from scientific literature (**Figure 2A**). LitQA2 questions are designed to have answers that appear in the main body of a paper, but not in the abstract, and ideally appear only once in the set of all scientific literature. These constraints enable us to evaluate response accuracy by matching the system's cited source DOI with the DOI originally assigned by the question creator. To enforce these criteria, we generated large numbers of questions about obscure intermediate findings from very recent papers, and then excluded any questions where either an existing AI system or a human annotator could answer the question using an alternative source (Section 8.2.1). These were generated entirely by experts, although there are emerging ideas about how to automate this process[12]. When

answering LitQA2 questions, models can refuse to answer via selecting *Insufficient information to answer this question*. Similar to prior work[13] and matching actual scientific questions, some questions are intended to be unanswerable. We evaluate two metrics: precision, the fraction of questions answered correctly when a response is provided, and accuracy, the fraction of correct answers over all questions. We also consider recall, which is the total percentage of questions where the system attributed its answer to the correct source DOI denoted in LitQA2.

Having developed LitQA2, we then utilized it to design an AI system for the scientific literature. The current paradigm for eliciting factually-based responses from LLMs is to use retrieval-augmented generation (RAG)[14,15]. RAG provides additional context to the LLM (e.g., snippets from research papers) to ground the generated response. As scientific literature is quite large, identifying the correct snippet is a challenge. Strategies like using metadata or hierarchical indexing can improve retrieval in this setting[16], but finding the correct paper for a task often requires iterating and revising queries. Inspired by PaperQA[17], PaperQA2 is a RAG *agent* that treats retrieval and response generation as a multi-step agent task[18] instead of a direct procedure. PaperQA2 decomposes RAG into *tools*, allowing it to revise its search parameters and to generate and examine candidate answers before producing a final answer (**Figure 1A**). PaperQA2 has access to a "Paper Search" tool, where the agent model transforms the user request into a keyword search that is used to identify candidate papers. The candidate papers are parsed into machine readable text, and chunked for later usage by the agent. PaperQA2 uses the state-of-the-art document parsing algorithm (Grobid[19]) that reliably parses sections, tables, and citations from papers. After finding candidates, PaperQA2 can use a "Gather Evidence" tool that first ranks paper chunks with a top-k dense vector retrieval step, followed by an LLM reranking and contextual summarization (RCS) step. RCS prevents irrelevant chunks from appearing in the RAG context by summarizing and scoring the relevance of each chunk, which is known to be critical for RAG[20]. The top ranked contextual summaries are stored in the agent's state for later steps. PaperQA2's design differs from similar RAG systems like Perplexity[21], Elicit[22], or Mao et al.[23] which deliver retrieved chunks without substantial transformation in the context of the user query. While RCS is more costly than retrieval without a contextual summary, it allows PaperQA2 to examine much more text per user question. The RCS step also injects metadata about the source paper, like its citation count and journal. Once the PaperQA2 state has summaries, it can call a "Generate Answer" tool which uses the top ranked evidence summaries inside a prompt to an LLM for the final response to the asked questions or assigned task. To further improve recall, PaperQA2 adds a new "Citation Traversal" tool (Section 8.1.1) that exploits the citation graph as a form of hierarchical indexing to add additional relevant sources.

In answering LitQA2 questions, PaperQA2 parsed and utilized an average of 14.5 ± 0.6 (mean $\pm$ SD, $n = 3$) papers per question. Running PaperQA2 on LitQA2 yielded a precision of $85.2\% \pm 1.1\%$ (mean $\pm$ SD, $n = 3$), and an accuracy of $66.0\% \pm 1.2\%$ (mean $\pm$ SD, $n = 3$) (**Figure 2B**), with the system choosing "insufficient information" in $21.9\% \pm 0.9\%$ (mean $\pm$ SD, $n = 3$) of answers. To compare the performance of PaperQA2 to other retrieval systems, we evaluated the performance of PaperQA with original parameters, commercial systems like Perplexity[21] and Elicit[22], and frontier (non-RAG models) on LitQA2. We found that PaperQA2 outperforms other RAG systems on the LitQA2 benchmark in both precision and accuracy. We also found that all RAG systems tested, with the exception of Elicit, outperform non-RAG frontier models in both precision and accuracy.

To ensure that we did not overfit PaperQA2 to achieve high performance on LitQA2, we generated a new set of 101 LitQA2 questions after making most of the engineering changes to PaperQA2. The accuracy of PaperQA2 on the original set of 147 questions did not differ significantly from its accuracy on the latter set of 101 questions, indicating that our optimizations in the first stage generalized well to new and unseen LitQA2 questions (Table 2).

To compare PaperQA2 performance to human performance on the same task, human annotators who either possessed a PhD in biology or a related science, or who were enrolled in a PhD program (see Section 8.2.1), were each provided a subset of LitQA2 questions and a performance-related financial incentive of $3-12 per question to answer as many questions correctly as possible within approximately one week, using any online tools and paper access provided by their institutions. Under these conditions, human annotators achieved $73.8\% \pm 9.6\%$ (mean $\pm$ SD, $n = 9$) precision on LitQA2 and $67.7\% \pm 11.9\%$ (mean $\pm$ SD, $n = 9$) accuracy (**Figure 2A**, green line). PaperQA2 thus achieved superhuman precision on this task ($t(8.6) = 3.49, p = 0.0036$) and did not differ significantly from humans in accuracy ($t(8.5) = -0.42, p = 0.66$).

3 Performance analysis of PaperQA2

We varied the parameters of PaperQA2 to understand which are responsible for its accuracy (**Figure 2C**). We created a non-agentic version (*No Agent*) which had a hard-coded sequence of actions (paper search, gather evidence, then generate answer). The non-agentic system had significantly lower accuracy ($t(3.7) = 3.41, p = 0.015$), validating the choice of using an agent. We attribute the performance difference to the agent's better recall because it can return to and change keyword searches (paper search tool calls) after observing the amount of relevant papers it finds. The highest

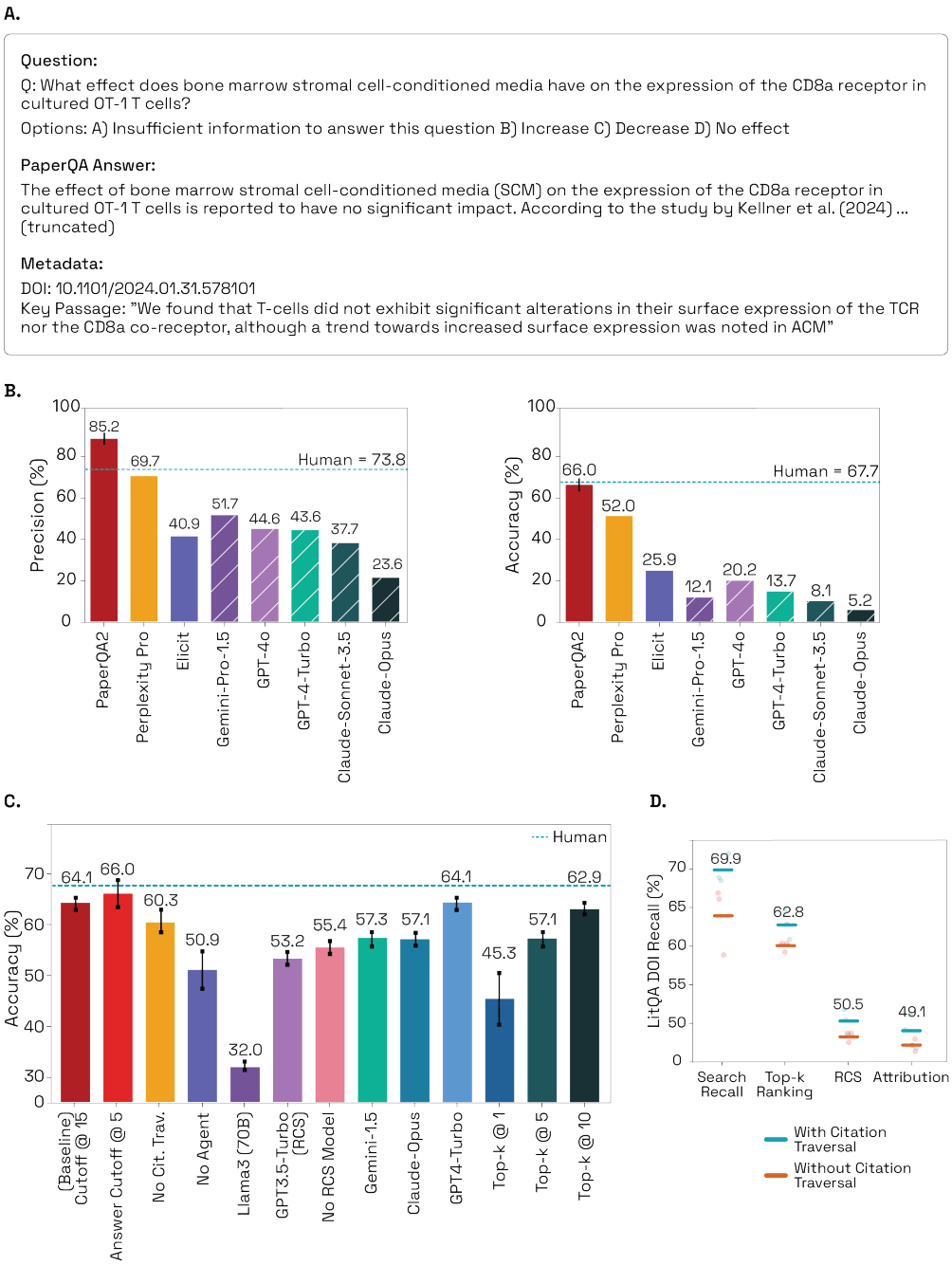


Figure 2: **A.** Example LitQA2 question, PaperQA2 answer, and metadata. **B.** PaperQA2 performance on LitQA2 across technologies. **C.** PaperQA2 performance studies and ablations across component categories. Error bars are 95% CI. **D.** Aggregated LitQA2 DOI recall per PaperQA2 stage. *Search Recall* includes DOIs found via the "Paper Search" or "Citation traversal" tools, *Top-k Ranking* includes all DOIs with similarity rankings below the top-k ranking cutoff (30), *RCS* includes all DOIs selected by RCS, and *Attribution* includes all DOIs cited in the "Generate Answer" tool.

accuracy LitQA2 runs had 1.26 ± 0.07 (mean $\pm$ SD) searches per question, and 0.46 ± 0.02 (mean $\pm$ SD) citation traversals per question showing that the agent will sometimes return to an additional search or traverse the citation graph to gather more papers.

We then explored the tools and their parameters available to PaperQA2. An important RAG parameter is the number of contexts or text chunks to include in the final text generation step ("Generate Answer" tool in **Figure 1A**). This is a balance because increasing the count improves the chance of including the key context needed to answer a question

(improving accuracy), but also increases the amount of distracting irrelevant context that reduces precision[24]. We varied the amount of contexts from 15 to 5 in **Figure S5** and **Figure 1** to see this effect; 15 gives highest precision and 5 gives highest accuracy.

To improve relevant chunk retrieval, we hypothesized that papers found as either citers or citees of existing relevant chunks would be an effective form of hierarchical indexing. This was validated by ablating the "Citation Traversal" tool (*No Cit. Trav.*), which showed an increased accuracy ($t(2.55) = 2.14, p = 0.069$), and significantly increased DOI recall ($t(3) = 3.4$, $p = 0.022$) at all stages of the PaperQA2 flow. (**Figure 1D**) This tool's process mirrors the way that scientists interact with the literature.

To quantify retrieval accuracy changes across LLM implementations, we performed experiments which varied the model choice for our "Generate Answer" tool as well as the RCS step in our "Gather Evidence" tool. Both of these tools give LLMs the opportunity to correctly identify crucial information in our scientific corpus, and we wanted to evaluate if their combined usage would be more effective than an un-transformed insertion of the top chunks into the final context window. The *No RCS Model* ablation validates that adding RCS to a traditional RAG text generation step significantly increases retrieval accuracy ($t(3.92) = 9.29, p < 0.001$). Interestingly, this is not true across all models tested, smaller models (GPT-3.5-Turbo (RCS), Llama3 (70B)) decrease overall accuracy when used for RCS, relative to not using a model at all. This indicated there is a comprehension threshold that must be met for effective summarization and relevance evaluation. GPT-4-Turbo significantly outperformed other models on LitQA2 accuracy when used in the RCS step ($t(3.47) = 6.14, p = 0.003$). Finally, Claude-Opus[25] had the highest LitQA2 precision (**Figure 5**), though it was not a significant increase over Gemini-1.5-Pro and GPT-4-Turbo.

To reduce the large cost of generating contextual summaries, we examined the effect of lowering the top-k ranking depth, i.e. limiting the number of passages considered in the RCS step. We see a significant increase in accuracy with increasing depth (i.e. more document chunks entering into the RCS ranking) from 1 to 10 ($t(2.15) = 5.44, p = 0.014$), and a long tail of diminishing performance gains from 10 to 30 (the default). In summary, having a deep (>10) RCS ranking list, with a high-performing LLM, was crucial in achieving human-level accuracy on LitQA2.

We had hypothesized that parsing quality would affect accuracy, but Grobid parsings and larger chunk sizes did not significantly increase precision, accuracy, or recall on LitQA2 (Figure 6). This is likely specific to being a retrieval task, as there is often only a single passage needed from a paper's body, which makes our result insensitive to parser changes. Anecdotally, we found better parsings to be crucial for extracting data from tables in WikiCrow (detailed in the Section 8.3).

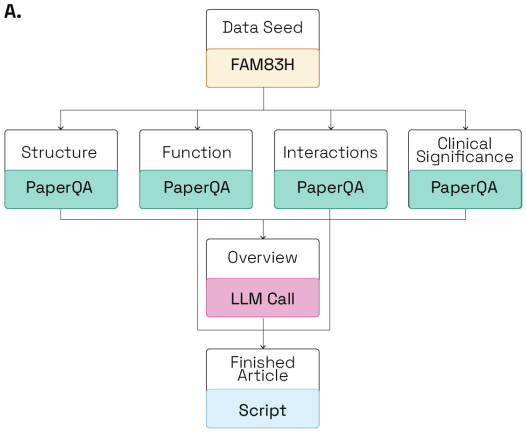
4 Summarizing scientific topics

To evaluate PaperQA2 on summarization, we engineered a system called WikiCrow, which generates cited Wikipedia-style articles about human protein-coding genes by combining several PaperQA2 calls on topics such as the structure, function, interactions, and clinical significance of the gene (**Figure 3A**). There has been previous work on *unconstrained* document summarization, where the document must be found and then summarized,[26] and even writing Wikipedia-style articles with RAG[2]. These studies have not compared directly against Wikipedia with human evaluation. Instead, they used either LLMs to judge or compared ROGUE (text overlap) against ground-truth summaries. Here, we measure directly against human-generated Wikipedia with subject mater expert grading.

We used WikiCrow to generate 240 articles on genes that already have non-stub Wikipedia articles to have matched comparisons. WikiCrow articles averaged 1219.0 ± 275.0 words (mean $\pm$ SD, $N = 240$), longer than the corresponding Wikipedia articles (889.6 ± 715.3 words). The average article was generated in 491.5 ± 324.0 seconds, and had an average cost of $\$4.48 \pm \$1.02$ per article (including costs for search and LLM APIs). We compared WikiCrow and Wikipedia on 375 statements sampled from the 240 paired articles. Statements were selected using cues from document formatting (Section 8.3). The initial article sampling excluded any Wikipedia articles that were "stubs" or incomplete articles. Statements were then shuffled and given, blinded, to human experts, who graded statements according to whether they were (1) cited and supported; (2) missing a citation; or (3) cited and unsupported. We found that WikiCrow had significantly fewer "cited and unsupported" statements than the paired Wikipedia articles (13.5% vs. 24.9%) ($p = 0.0075, \chi^2(1), N = 375$ for all tests in this section). WikiCrow failed to cite sources at a 3.9x lower rate than human written articles, as only 3.5% of WikiCrow statements were uncited, vs. 13.6% for Wikipedia ($p < 0.001$). In addition, defining precision for WikiCrow as the ratio of cited and supported statements over all cited statements, we found that WikiCrow displayed significantly higher precision than human-written articles (86.1% vs. 71.2%, $p = 0.0013$).

The "cited and unsupported" evaluation category includes both inaccurate statements (e.g. true hallucinations or reasoning errors) and statements that are accurate with inappropriate citations. To investigate the nature of the errors

A.



C.

	WikiCrow	Wikipedia
Average Article Length (words)	1,219	890
Total Statements Graded	189	186
Valid Statements	171	169
Cited and Supported (Accuracy)	83.0%	61.5%
Uncited	3.5%	13.6%
Cited and Unsupported	13.5%	24.9%
Supported Among Cited (Precision)	86.1%	71.2%

D.

	WikiCrow	Wikipedia
Unsupported Breakout (total)	23	42
Reasoning Issues	12	26
Attribution Issues	10	16
Trivial Statements	1	0

B.

FAM83H

Overview
FAM83H, or Family with Sequence Similarity 83 Member H, is a gene located on chromosome 8 in humans that encodes a protein belonging to the FAM83 protein family. This protein family is characterized by a conserved domain known as DUF1669, which includes a pseudo-phospholipase D motif, although it lacks catalytic activity, suggesting a role as a pseudoenzyme (Bozatzi2018The)...

Structure
The molecular structure of the FAM83H protein is characterized by a conserved N-terminal domain known as DUF1669, which includes a pseudo-phospholipase D (PLD) catalytic motif. This motif, however, lacks essential elements for actual catalytic activity, rendering the FAM83 proteins as pseudoenzymes (Bozatzi2018The). The primary structure of FAM83H lacks well-defined structural characteristics or known functional domains, except for the mentioned PLD-like domain, suggesting a conserved secondary structure and similar three-dimensional fold (Wang2015Fam83h)...

Function
FAM83H, or Family with Sequence Similarity 83 Member H, plays a crucial role in the regulation of keratinization in epithelial cells, which is essential for the formation of the protective outer layer of the skin (Barr2019Long). This gene is also involved in the organization of the keratin cytoskeleton, maintaining the structural integrity and function of epithelial tissues (Barr2019Long)...

Clinical Significance
Mutations in the FAM83H gene are primarily linked to autosomal dominant hypocalcified amelogenesis imperfecta (ADHCAI), a genetic disorder that affects dental enamel formation, leading to enamel that is soft, thin, and prone to rapid abrasion after tooth eruption. The condition is characterized by yellowish-brown discolored enamel that exhibits reduced mineral density and increased sensitivity (Sriwattanapong2021Age-related; Lee2010FAM83H)...

Interactions
FAM83H interacts with various proteins, playing a significant role in cellular processes such as keratin filament organization and signaling pathways. It binds to casein kinase Iα (CK-1α) through a specific motif in its N-terminal region, which is crucial for recruiting CK-1α to keratin filaments in epithelial cells. This interaction is essential for the dynamic rearrangement of keratin filaments, impacting cell migration and polarization, particularly in colorectal cancer cells (Kuga2013A)...

Figure 3: **A.** WikiCrow prompt graph, showing the initial seed variable (gene name), PaperQA2 prompts for each section, and overview LLM call. Each section feeds into a Python script which stitches the outputs together. **B.** Sample WikiCrow article for gene FAM83H, with truncated sections under each header. **C.** Performance statistics for a WikiCrow vs. Wikipedia comparison performed by evaluators. **D.** "Cited and unsupported" issue categorization counts.

in Wikipedia and WikiCrow further, we manually inspected all reported errors and attempted to classify the issues as follows: reasoning issues, i.e. the written information contradicts, over-extrapolates, or is unsupported by any included citations; attribution issues, i.e. the information is likely supported by another included source, but either the statement does not include the correct citation locally or the source is too broad (e.g. a database portal link); or trivial statements, which are true passages, but overly pedantic or unnecessary (**Figure 3D**). Surprisingly, we found that compared to Wikipedia, WikiCrow had significantly fewer reasoning errors (12 vs. 26, $p = 0.0144$, $\chi^2(1), N = 375$) but a similar number of attribution errors (10 vs. 16, $p = 0.21$), suggesting that the improved factuality of WikiCrow over Wikipedia was largely due to improvements in reasoning. Although language models are clearly prone to reasoning errors (or hallucinations), in our task at least they appear to be less prone to such errors than Wikipedia authors or editors. This

statement is specific to the agentic RAG setting presented here: language models like GPT-4 on their own, if asked to generate Wikipedia articles, would still be expected to hallucinate at high rates. [a]

5 Detecting contradictions in the literature

Because PaperQA2 can explore scientific literature at much higher throughput than human scientists, we reasoned that we could deploy it to systematically identify contradictions and inconsistencies in the literature at scale. Contradiction detection is a "one versus many" problem, which in principle involves comparing claims or statements in one paper with all other claims or statements in the literature. At scale, contradiction detection becomes a "many versus many" problem and loses feasibility for humans. Thus, we leveraged PaperQA2 to build a system called ContraCrow that automatically detects contradictions in the literature (**Figure 4A**).

Contradiction detection is also known as claim verification or colloquially as "fact checking"[27]. This task has been studied for over a decade, especially in the context of claims in the news or the internet[27,28]. Although originally restricted to context and a claim, the setting extended to be unconstrained[29] and recent work tries to work at the scale of the internet[30]. Some claim verification work has also focused on scientific claims[31,32]. The main novelty of this work is in detecting contradictions without a restricted corpus and evaluating with human experts, not against a benchmark.

ContraCrow first extracts claims from a provided paper using a series of LLM completion calls (similar to Schlichtkrull et al. (2024)[30]), and then feeds those claims into PaperQA2 with a *contradiction detection* prompt. This prompt instructs the system to evaluate whether there are any contradictions in literature to the provided claim, providing both an answer and a choice from an 11-point Likert scale (**Figure 4B**, Methods Section 8.4). Utilizing a Likert Scale allows the system to give more reliable and interpretable scores when providing rank[33].

To evaluate ContraCrow, we first derived a contradiction detection benchmark, ContraDetect, from LitQA2, as detailed in Section 8.4.2. Briefly, we converted half of the question-answer pairs in LitQA2 into declarative, incorrect statements that are contradicted by the papers referenced in the corresponding LitQA2 question. (For example, a question like "what color is grass?" would become "grass is purple.") We converted the other half of the LitQA2 questions into declarative, correct statements that are supported by the corresponding papers ("what color is grass" becomes "grass is green.").

We then evaluated ContraCrow on its ability to detect the contradictions in ContraDetect. By transforming the Likert scale output into integers, we were able to tune the detection threshold and obtain an ROC curve with an AUC of 0.842 (**Figure 4C**). Setting a threshold of 8 (*contradiction*), ContraCrow achieved 73% accuracy, 88% precision, and a false positive rate of only 7%. To further evaluate precision and to demonstrate ContraCrow's ability to handle "Has anybody ever done x?" questions, we then ran ContraCrow on 42 "no-evidence statements" that have never to our knowledge been reported on (detailed in Section 8.4.2). These claims were hand-generated by the authors within their fields of expertise, and ContraCrow correctly chose *lack of evidence* (5) as its response 98% of the time, indicating its ability to distinguish between real contradictions and lack of support.

We then applied ContraCrow to a set of 93 biology-related papers randomly selected from our database, identifying an average of 35.16 ± 21.72 (mean $\pm$ SD, $N = 93$) claims per paper. Of the 3,180 claims analyzed over the 93 papers, 6.85% were deemed by ContraCrow to be contradicted by the literature, with 2.89%, 3.77%, and 0.19% assigned scores 8, 9, and 10, respectively (**Figure 4D**). Setting a Likert scale threshold of 8, we detected an average of 2.34 ± 1.99 contradictions per paper (mean $\pm$ SD) (**Figure 4E**). As an example, one contradiction detected by our system concerned the prognostic implications of LEF1 expression in colorectal carcinomas. The source paper, Kriegl et al. (2010)[34], finds using immunohistochemistry on a tissue microarray that LEF-1 correlates positively with longer overall survival. By contrast, a study published the following year also using tissue microarrays found LEF1 overexpression in colorectal cancer correlates negatively with longer overall survival and is also correlated with liver metastasis[35], results that have been supported by other studies as well[36,37,38], thus explicitly contradicting the original statement.

To evaluate the validity of the contradictions detected this way, expert human annotators evaluated 50 claims that were assigned ContraCrow scores of 8 and 50 claims that were assigned scores of 9 or 10, considering all evidence and reasoning cited by the model. Annotators agreed with ContraCrow's findings on 70% of evaluated claims, or 1.64 contradictions per paper, demonstrating significant agreement ($p = 1e^{-4}$, $\chi^2(1)$), with an F1 score of 0.82. Interestingly, claims scored as 8 by ContraCrow were not more likely to elicit human agreement (70%) than claims scored as 9 or 10 (70%) (**Figure 4F**). The final number, 1.64 validated contradictions per paper, serves as a lower bound on the abundance of human-validatable contradictions in the biology literature.

[a] All WikiCrow generated articles for this study are now available for download from a Google Cloud Storage bucket `https://storage.googleapis.com/fh-public/wikicrow2/`. A command line tool like `gsutil` can be used to list and bulk-access these files. All WikiCrow and LitQA evaluator responses are available via the following link.

Figure 4: **A.** Schematic of ContraCrow. **B.** Likert scale used to evaluate contradictions, along with the integer mapping. **C.** ROC curve and metrics (Likert scale threshold of 8) of ContraCrow's performance on ContraDetect benchmark. **D.** Bar plot showing percentage of all claims found in 93 papers that are given scores of 8, 9, and 10 by ContraCrow. **E.** Histogram of the number of contradictions found per paper over 93 papers. The table shows the average number of claims (35.16 ± 21.72 [mean $\pm$ SD, $N = 93$]) and contradictions (2.34 ± 1.99 [mean $\pm$ SD, $N = 93$]) per paper found. **F.** Proportion of ContraCrow contradictions with scores 8 and 9-10 validated by expert evaluators. **G.** Range of evaluator scores for each claim. Bars represent the range of Likert scale scores assigned by expert evaluators and stars represent the ContraCrow label on the same claims. The table shows a mean standard deviation per claim of 1.45 ($N = 30$ claims) over all evaluators and a mean Pearson correlation coefficient of 0.579 ± 0.197 (mean $\pm$ SD) between evaluators ($N = 5$).

We were concerned that the annotators on the "contradiction validation" task might exhibit some bias toward agreeing with the model or otherwise be influenced by the model's reasoning and chosen sources. We therefore further evaluated ContraCrow by asking human experts to identify contradictions in the same literature considered by ContraCrow, without any access to ContraCrow's reasoning. This "contradiction detection" task (Section 8.4.2) is a much more challenging task for annotators than the contradiction validation task, because it requires them to exhaustively consider a significant amount of literature relevant to the claim in question, whereas the contradiction validation task only requires them to consider the specific sections cited by the model. In the contradiction detection task, human experts were provided with the top 15 paper chunks identified as relevant to the claim by PaperQA2 (the same chunks ContraCrow had access to), and were asked to respond whether the claim was contradicted by the available evidence on both the Likert scale and binary determination ("yes" or "no"). Experts considered a mixture of claims rated as contradictions and non-contradictions by ContraCrow. We found that experts' binary responses agree with each other on $75.5\% \pm 13.43\%$

of claims provided (mean $\pm$ SD, $N = 10$ pairwise comparisons between human annotators), whereas they agree with ContraCrow on $60.42\% \pm 5.99\%$ of claims (mean $\pm$ SD, $N = 5$ human annotators), indicating that humans are significantly more correlated with each other than they are with ContraCrow ($p = 0.015$). Examining the Likert scale values over 30 claims evaluated by human annotators leads us to speculate that overconfidence on ContraCrow's part is the primary driver of its lack of agreement with human annotators, and indicates directions for future improvement (**Figure 4G**).

We are thus able to establish a lower bound for the average number of human-validatable contradictions per paper in biology literature at 1.64. Importantly, just because a claim is detected as a contradiction does not mean that the claim is wrong. For example, one detected contradiction is *"GBP is only found in the cytosol of human fibroblasts."*[39] This example shows how scientific literature and findings can update and contradict over time, as ContraCrow points out: *"...more recent research has shown that GBPs can localize to various cellular compartments and membranes*[40,41,42,43]*."* Thus, claims can contradict but still be contextually valid, and contradictory claims may be merely reflective of the iterative nature of research.

6 Conclusions

We developed a methodology to compare or validate AI systems against human performance in realistic tasks for scientific research. PaperQA2 outperforms human experts on answering questions across all scientific literature; produces summaries that are, on average, more factual than Wikipedia summaries; and can be deployed to identify contradictions in scientific literature at scale. The contradiction work, in particular, attests to the potential of systems such as PaperQA2 for science: notably, one human expert who performed this task, who was also tested on a large battery of other benchmarks, reported without solicitation that the contradiction detection task was the hardest task they were asked to perform. Although PaperQA2 is expensive compared to lower accuracy commercial systems, it is inexpensive in absolute terms, costing $1 to $3 per query. Scaling up PaperQA2 and other literature-enabled agents like WikiCrow and ContraCrow empowers us to take advantage of the latent knowledge in literature at much greater scale than is possible today.

7 Data Availability

The code necessary to replicate these results or modify the algorithm for further research is included on Github via paperqa. Data including all evaluator responses, contradiction detection claims, litQA questions, and WikiCrow candidate statements are available in the supplementary materials. All generated WikiCrow articles for this study are available in a public Google Cloud bucket here: `https://storage.googleapis.com/fh-public/wikicrow2/`.

Author Contributions MDS led work on PaperQA2 and WikiCrow, in collaboration with JDB. SC led work on ContraCrow. JML led work on LitQA. MH, MJH, and MP contributed questions to the LitQA benchmark. SGR and ADW conceived of and supervised the project.

Acknowledgments Work at FutureHouse is supported by the generosity of Eric and Wendy Schmidt. We also acknowledge Matt Rubashkin for contributions to an earlier stage of the project, and we acknowledge all current members of FutureHouse for useful discussions, including Sid Narayanan, Ryan Rhys-Griffiths, Cade Gordon, Peter Chang, and Conor Igoe. We also acknowledge support and resources from the Semantic Scholar Project at the Allen Institute for AI.

Competing interests The authors report no competing interests. FutureHouse, Inc. is a non-profit research organization.

References

[1] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2023.

[2] Yijia Shao, Yucheng Jiang, Theodore A Kanell, Peter Xu, Omar Khattab, and Monica S Lam. Assisting in writing wikipedia-like articles from scratch with large language models. *arXiv preprint arXiv:2402.14207*, 2024.

[3] Kyle Lo, Joseph Chee Chang, Andrew Head, Jonathan Bragg, Amy X. Zhang, Cassidy Trier, Chloe Anastasiades, Tal August, Russell Authur, Danielle Bragg, Erin Bransom, Isabel Cachola, Stefan Candra, Yoganand Chandrasekhar, Yen-Sung Chen, Evie (Yu-Yen) Cheng, Yvonne Chou, Doug Downey, Rob Evans, Raymond

Fok, F.Q. Hu, Regan Huff, Dongyeop Kang, Tae Soo Kim, Rodney Michael Kinney, Aniket Kittur, Hyeonsu B Kang, Egor Klevak, Bailey Kuehl, Michael Langan, Matt Latzke, Jaron Lochner, Kelsey MacMillan, Eric Stuart Marsh, Tyler C. Murray, Aakanksha Naik, Ngoc-Uyen Nguyen, Srishti Palani, Soya Park, Caroline Paulic, Napol Rachatasumrit, Smita R Rao, Paul Sayre, Zejiang Shen, Pao Siangliulue, Luca Soldaini, Huy Tran, Madeleine van Zuylen, Lucy Lu Wang, Christopher Wilhelm, Caroline M Wu, Jiangjiang Yang, Angele Zamarron, Marti A. Hearst, and Daniel S. Weld. The semantic reader project: Augmenting scholarly documents through ai-powered interactive reading interfaces. *ArXiv*, abs/2303.14334, 2023.

[4] SM Tonmoy, SM Zaman, Vinija Jain, Anku Rani, Vipula Rawte, Aman Chadha, and Amitava Das. A comprehensive survey of hallucination mitigation techniques in large language models. *arXiv preprint arXiv:2401.01313*, 2024.

[5] Matthew Dahl, Varun Magesh, Mirac Suzgun, and Daniel E Ho. Large legal fictions: Profiling legal hallucinations in large language models. *Journal of Legal Analysis*, 16(1):64–93, 2024.

[6] Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. PubMedQA: A dataset for biomedical research question answering. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *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, November 2019. Association for Computational Linguistics.

[7] Anastasia Krithara, Anastasios Nentidis, Konstantinos Bougiatiotis, and Georgios Paliouras. Bioasq-qa: A manually curated corpus for biomedical question answering. *Scientific Data*, 10(1):170, 2023.

[8] Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William W Cohen, and Xinghua Lu. Pubmedqa: A dataset for biomedical research question answering. *arXiv preprint arXiv:1909.06146*, 2019.

[9] Shuai Wang, Yi He, Zi Chen, Yanzhang Li, Jingping Zhao, and Luxian Lyu. Pleiotropic action of genetic variation in znf804a on brain structure: a meta-analysis of magnetic resonance imaging studies. *Neuropsychiatric Disease and Treatment*, pages 721–729, 2019.

[10] Qinling Wei, Meng Li, Zhuang Kang, Leijun Li, Feici Diao, Ruibin Zhang, Junjing Wang, Liangrong Zheng, Xue Wen, Jinbei Zhang, et al. Znf804a rs1344706 is associated with cortical thickness, surface area, and cortical volume of the unmedicated first episode schizophrenia and healthy controls. *American Journal of Medical Genetics Part B: Neuropsychiatric Genetics*, 168(4):265–273, 2015.

[11] Jon M. Laurent, Joseph D. Janizek, Michael Ruzo, Michaela M. Hinks, Michael J. Hammerling, Siddharth Narayanan, Manvitha Ponnapati, Andrew D. White, and Samuel G. Rodriques. Lab-bench: Measuring capabilities of language models for biology research. *arXiv preprint arXiv:2407.10362*, 2024.

[12] Yuwei Wan, Aswathy Ajith, Yixuan Liu, Ke Lu, Clara Grazian, Bram Hoex, Wenjie Zhang, Chunyu Kit, Tong Xie, and Ian Foster. Sciqag: A framework for auto-generated scientific question answering dataset with fine-grained evaluation. *arXiv preprint arXiv:2405.09939*, 2024.

[13] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*, 2018.

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

[15] Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. Retrieval augmentation reduces hallucination in conversation. *arXiv preprint arXiv:2104.07567*, 2021.

[16] Yunfan Gao, Tao Sheng, Youlin Xiang, Yun Xiong, Haofen Wang, and Jiawei Zhang. Chat-rec: Towards interactive and explainable llms-augmented recommender system. *arXiv preprint arXiv:2303.14524*, 2023.

[17] Jakub Lála, Odhran O'Donoghue, Aleksandar Shtedritski, Sam Cox, Samuel G Rodriques, and Andrew D White. Paperqa: Retrieval-augmented generative agent for scientific research. *arXiv preprint arXiv:2312.07559*, 2023.

[18] Ehud Karpas, Omri Abend, Yonatan Belinkov, Barak Lenz, Opher Lieber, Nir Ratner, Yoav Shoham, Hofit Bata, Yoav Levine, Kevin Leyton-Brown, et al. Mrkl systems: A modular, neuro-symbolic architecture that combines large language models, external knowledge sources and discrete reasoning. *arXiv preprint arXiv:2205.00445*, 2022.

[19] Grobid. `https://github.com/kermitt2/grobid`, 2008–2023.

[20] Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H Chi, Nathanael Schärli, and Denny Zhou. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning*, pages 31210–31227. PMLR, 2023.

[21] Perplexity. https://www.perplexity.ai/. Accessed: 2024-07-10.

[22] Elicit. https://www.elicit.com/. Accessed: 2024-07-10.

[23] Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. Generation-augmented retrieval for open-domain question answering. *arXiv preprint arXiv:2009.08553*, 2020.

[24] Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.

[25] Introducing the next generation of Claude — anthropic.com. https://www.anthropic.com/news/claude-3-family. [Accessed 13-07-2024].

[26] John Giorgi, Luca Soldaini, Bo Wang, Gary D Bader, Kyle Lo, Lucy Lu Wang, and Arman Cohan. Exploring the challenges of open domain multi-document summarization. *ArXiv*, abs/2212.10526, 2022.

[27] Andreas Vlachos and Sebastian Riedel. Fact checking: Task definition and dataset construction. In *Proceedings of the ACL 2014 workshop on language technologies and computational social science*, pages 18–22, 2014.

[28] William Ferreira and Andreas Vlachos. Emergent: a novel data-set for stance classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: Human language technologies*. ACL, 2016.

[29] Kashyap Popat, Subhabrata Mukherjee, Jannik Strötgen, and Gerhard Weikum. Where the truth lies: Explaining the credibility of emerging claims on the web and social media. In *Proceedings of the 26th international conference on world wide web companion*, pages 1003–1012, 2017.

[30] Michael Schlichtkrull, Zhijiang Guo, and Andreas Vlachos. Averitec: A dataset for real-world claim verification with evidence from the web. *Advances in Neural Information Processing Systems*, 36, 2024.

[31] David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. Fact or fiction: Verifying scientific claims. *arXiv preprint arXiv:2004.14974*, 2020.

[32] David Wadden, Kyle Lo, Lucy Lu Wang, Arman Cohan, Iz Beltagy, and Hannaneh Hajishirzi. Multivers: Improving scientific claim verification with weak supervision and full-document context. *arXiv preprint arXiv:2112.01640*, 2021.

[33] Bhashithe Abeysinghe and Ruhan Circi. The challenges of evaluating llm applications: An analysis of automated, human, and llm-based approaches. *arXiv preprint arXiv:2406.03339*, 2024.

[34] Lydia Kriegl, David Horst, Jana A Reiche, Jutta Engel, Thomas Kirchner, and Andreas Jung. Lef-1 and tcf4 expression correlate inversely with survival in colorectal cancer. *Journal of translational medicine*, 8:1–8, 2010.

[35] Albert Y Lin, Mei-Sze Chua, Yoon-La Choi, William Yeh, Young H Kim, Raymond Azzi, Gregg A Adams, Kristin Sainani, Matt van de Rijn, Samuel K So, et al. Comparative profiling of primary colorectal carcinomas and liver metastases identifies lef1 as a prognostic biomarker. *PloS one*, 6(2):e16636, 2011.

[36] Wen-Juan Wang, Yu Yao, Li-Li Jiang, Ting-Hua Hu, Jie-Qun Ma, Zhi-Ping Ruan, Tao Tian, Hui Guo, Shu-Hong Wang, and Ke-Jun Nan. Increased lef1 expression and decreased notch2 expression are strong predictors of poor outcomes in colorectal cancer patients. *Disease markers*, 35(5):395–405, 2013.

[37] Wen-Juan Wang, Yu Yao, Li-Li Jiang, Ting-Hua Hu, Jie-Qun Ma, Zi-Jun Liao, Jun-Tao Yao, Dong-Fan Li, Shu-Hong Wang, and Ke-Jun Nan. Knockdown of lymphoid enhancer factor 1 inhibits colon cancer progression in vitro and in vivo. *PLoS One*, 8(10):e76596, 2013.

[38] Taher R Kermanshahi, Priya Jayachandran, Daniel T Chang, and Reet Pai. Lef-1 is frequently expressed in colorectal carcinoma and not in other gastrointestinal tract adenocarcinomas: an immunohistochemical survey of 602 gastrointestinal tract neoplasms. *Applied Immunohistochemistry & Molecular Morphology*, 22(10):728–734, 2014.

[39] YS Cheng, Mary F Becker-Manley, Theresa P Chow, and Deborah C Horan. Affinity purification of an interferon-induced human guanylate-binding protein and its characterization. *Journal of Biological Chemistry*, 260(29):15834–15839, 1985.

[40] Nathalie Britzen-Laurent, Michael Bauer, Valeria Berton, Nicole Fischer, Adrian Syguda, Simone Reipschläger, Elisabeth Naschberger, Christian Herrmann, and Michael Stürzl. Intracellular trafficking of guanylate-binding proteins is regulated by heterodimerization in a hierarchical manner. *PloS one*, 5(12):e14246, 2010.

[41] Nir Modiano, Yanping E Lu, and Peter Cresswell. Golgi targeting of human guanylate-binding protein-1 requires nucleotide binding, isoprenylation, and an ifn-γ-inducible cofactor. *Proceedings of the National Academy of Sciences*, 102(24):8680–8685, 2005.

[42] Elisabeth Naschberger, Clara Lubeseder-Martellato, Nadine Meyer, Ruth Gessner, Elisabeth Kremmer, Andrè Gessner, and Michael Stürzl. Human guanylate binding protein-1 is a secreted gtpase present in increased concentrations in the cerebrospinal fluid of patients with bacterial meningitis. *The American journal of pathology*, 169(3):1088–1099, 2006.

[43] Luca Schelle, João Vasco Côrte-Real, Pedro José Esteves, Joana Abrantes, and Hanna-Mari Baldauf. Functional cross-species conservation of guanylate-binding proteins in innate immunity. *Medical Microbiology and Immunology*, 212(2):141–152, 2023.

8 Methods

8.1 PaperQA Implementation and Parameters

All reported figures and data in this work were built on the open source PaperQA package, available on GitHub at paperqa. While the core PaperQA repository provides the basic algorithms used, it does not include the Grobid parsing code, access to non-local full-text literature searches, or the citation traversal tool. The open source version of PaperQA utilizes LangChain [1] for its agentic and state update operations. The full configuration objects for all experiments run in this paper are included for further customization.

Note that while `paperqa` gives the ability to recreate this work, the experiments reported in this paper were performed using a more featureful HTTP server that takes advantage of bespoke infrastructure at the authors' institution. This infrastructure includes features such as user authentication, MongoDB request caching, Redis object caching and global-load balancing, several PostgreSQL DBs with associated ORM code, cost-monitoring modules, time-profiling modules, configuration storage and run orchestration (Dagster [2] and kubernetes [3]), cloud bucket storage for PDFs, a CI pipeline with semi-automated deployments, and infrastructure code for deploying auto-scaling instances in the cloud. None of these features affect performance on a per-query basis, but provide increased scalability, measurability, and persistence. To run our same server infrastructure, users would need to provision all of these assets and configure the deployments themselves. `paperqa` should serve allow usage and customization sufficient for most research purposes, and should be sufficient to reproduce the results reported here.

Even within `paperqa`, the "Paper Search" tool is limited by access to full text repositories of scientific papers, often bound by licensing agreements. The included implementation only works from local files accessible to each user. Our implementation starts with a full or partial-text keyword search, where the keywords have been specified by the agent when selecting the paper search tool. The ranked results returned from these services are then matched to a user's existing paper repository or can be retrieved on-the-fly if open-access or partner links exist for these works. These matching papers are parsed, and pulled into our agent state for usage with other tools. Note that the search services will have access to a larger corpus of works than is available to us via our repository and accessible link traversal, in these cases the system will simply skip these papers and they are not used. A stub of the paper search tool is implemented in `paperqa` with directions for users to implement their own retrieval since it will be limited to their own access to full-text papers.

Ablations and configurations for workflows like WikiCrow are exposed in `paperqa` as nested configuration objects. All experiments performed in this work correspond to included configuration objects. Here we highlight the configuration variable descriptions corresponding to the salient features tested in this work, though all variable names are available in the included files.

- `query`: The main query task asked of the PaperQA agent, i.e. a LitQA question or a directive to write an article.
- `llm`: The LLM used in the generate answer tool can be a valid Anthropic, OpenAI, or Gemini model identifier. This parameter was varied for the model experiments in Figure 2.
- `agent_llm`: The LLM used for the agent orchestration, in this work, it was always fixed to `gpt-4-turbo-2024-04-09`.
- `summary_llm`: The LLM used for the RCS step in the gather evidence tool, must be a valid Anthropic, OpenAI, or Gemini model identifier. This parameter was varied for the model ablations in Figure 2.
- `prompts`: A `PromptCollection` object from PaperQA [4], which allows for specification of prompts in each tool, as well as features like turning off RCS (via `prompts.skip_summarization`). The *No RCS Model* ablation used this input as well as the WikiCrow prompts.

- `max_sources`: The number of top ranked sources to be included in the generate answer tool, in **Figure 1A**, the 'filter top summaries" cutoff. This parameter was 5 in the top-performing *Answer cutoff @ 5*, but 15 for all other experiments.
- `consider_sources`: The top-k cutoff, i.e. the number of chunks that will be used in the RCS step. This parameter was set to 30 by default in LitQA experiments, save for the *Top-k Rank @ X* experiment where it was set to X. Additionally, for our WikiCrow prompts this parameter was set to 25.
- `agent_tools`: An ordered list of tool names that will be used by the agent, including `gather_evidence`, `paper_search`, `generate_answer`, and `citations_traversal`. This always included all four tools except for the *No Cit. Trav.* and *No Agent* runs where `citations_traversal` was excluded.
- `docs_index_mmr_lambda`: A pre-gather evidence MMR lambda parameter which can be used to pre-filter similar papers by name before gathering evidence. This was set to 0.9 for our WikiCrow run to promote diversity of sources, but 1.0 for LitQA experiments.
- `parsing_configuration.ordered_parser_preferences`: A list of the parsing algorithm to use, either `paperqa_default` (PyMuPDF) or `grobid`. `paperqa_default` was the default for each ablation, and `grobid` was used for WikiCrow generation. This parameter was also varied in the experiments shown in **Figure 6**.
- `parsing_configuration.chunksize`: the chunk size (in characters) to be used when chunking parsed documents. This parameter was varied in the experiments shown in **Figure 6**.
- `parsing_configuration.overlap`: the overlap (in characters) that will be common between sequential chunks. This was fixed at 750 for this work.
- `parsing_configuration.chunking_algorithm`: the algorithm used to chunk documents, `simple_overlap` simply uses a sliding window with overlap, and `sections` uses semantic parsing by section (i.e. one chunk per section where possible), if sections need to be broken into multiple chunks the system will automatically handle this. `sections` is only supported via `parsing_configuration.ordered_parser_preferences=grobid`. This parameter was varied in the experiments shown in Figure 6, and in our WikiCrow generation.
- `temperature`: temperature used for the LLM in the generate answer tool. This was set to 0 for all runs in this work.
- `summary_temperature`: temperature used for the LLM in the gather evidence tool's RCS step, this was set to 0 for all runs in this work.

8.1.1 Tool implementations

PaperQA2's agentic tools were implemented as in PaperQA[4]. Our agent was prompted with the following message to guide tool usage:

> Answer question: {question}. Search for papers, gather evidence, collect papers cited in evidence then re-gather evidence, and answer. Gathering evidence will do nothing if you have not done a new search or collected new papers. If you do not have enough evidence to generate a good answer, you can:
> - Search for more papers (preferred)
> - Collect papers cited by previous evidence (preferred)
> - Gather more evidence using a different phrase
>
> If you search for more papers or collect new papers cited by previous evidence, remember to gather evidence again. Once you have five or more pieces of evidence from multiple sources, or you have tried a few times, call {gen_answer_tool_name} tool. The {gen_answer_tool_name} tool output is visible to the user, so you do not need to restate the answer and can simply terminate if the answer looks sufficient. The current status of evidence/papers/cost is {status}

Where variables like {status} are included to represent the current state to the agent. Tools were implemented with the following prompts and settings.

Paper Search Tool

The paper search tool uses an initial keyword search, generated by the agent in the context of the user query. The agent is prompted as follows:

> A search query in this format: [query], [start year]-[end year]. You may include years as the last word in the query, e.g. 'machine learning 2020' or 'machine learning 2010-2020'. The current year is {get_year()}. The query portion can be a specific phrase, complete sentence, or general keywords, e.g. 'machine learning for immunology'.

Our initial search relies on services like Semantic Scholar[5], where candidate lists (default of 12) of relevant papers are generated then parsed. When parsed, the papers are first turned into text using either Grobid or PyMuPDF, then split into `chunksize` character sized pieces. If the `sections` parsing is used, then section chunks are split on header metadata provided by Grobid. An embedding vector is generated for each chunk using a hybrid implementation which concatenates a dense and sparse, keyword based embedding model. For the experiments included in this study, OpenAI's `text-embedding-large-3` was used. It was concatenated with a normalized 256 dimension vector which used modulus-encoding to extract a hot-encoded keyword from the tokenization integers provided by OpenAI's tiktoken[6] library. These text chunks are put into a document context which is accessible by the agent for further manipulation with tools. The PaperQA entrypoint for these functions can be found on github.

Gather Evidence Tool

As detailed in github for PaperQA, the Gather Evidence tool begins with a top-k vector ranking step, using the embedding vectors created in the Paper Search tool. The user query is embedded with the same algorithm, and cosine similarity is used to match all document chunks in the agent's context with the user query. The top-k chunks are then selected for the RCS step.

The reranking and contextual summarization step most differentiates PaperQA's implementation relative to other RAG technologies. The tool's prior step, an top-k vector retrieval ranking, is a widely implemented[7,8] approach to identify relevant documents, however, the RCS second step, is unique to PaperQA (to the authors' knowledge). While performance improvements with both deep reranking (or LLM) models and map-reduced summarizations[9,10,11,12] are well documented, combining the reranking operation with a contextual summary provides novel benefits.

The step is implemented by mapping an LLM completion across each top-k chunk (system prompt):

> Provide a summary of the relevant information that could help answer the question based on the excerpt. The excerpt may be irrelevant. Do not directly answer the question - only summarize relevant information. Respond with the following JSON format: {{ "summary": "...", "relevance_score": "..." }} where "summary" is relevant information from text - {summary_length} words and "relevance_score" is the relevance of "summary" to answer the question (integer out of 10)

Where each chunk is injected as follows:

> Excerpt from citation —- {text} —- Query: {question}

After completion, each JSON object is parsed and the passages are re-ranked according to the new relevance scores. When running with WikiCrow, gene names are also prompted to be extracted as additional JOSN keys, these are kept and injected in the final answering context. Advantages of the RCS step are as follows: 1. Token usage efficiency is vastly improved, a contextual summary will be 200-400 tokens compared with our standard document's chunk size of 2,250 tokens. This allows for a significantly more accessible document corpus for injection into PaperQA's answering context window. Furthermore, we see no decrease in summarization efficacy, using LitQA performance as a proxy, across document chunk sizes from 750-3,000 tokens. 2. As a new feature in this work, the LLM can be prompted to provide its summary in a structured JSON or XML format to simplify its downstream data extraction. In addition to a relevance score used for reranking, this structure can include metadata (such as a gene name) which will be retained through the PaperQA workflow. This is used to reduce hallucination and confusion in the final answer context. Since the RCS step is performed in an embarrassingly parallel fashion, it's highly efficient, and its utility can be applied to an arbitrarily deep ranking, up to the rate or cost limits of the LLM API. Our studies on the efficacy of the RCS depth led us to use a much deeper RCS depth, and to utilize the best performing model for the RCS operation. This differs from the intuition in prior work[4], which utilized a cheaper model during the RCS step.

Generate Answer Tool

This tool answers questions by taking a subset of the top ranked sources (from the RCS ranking), and injects them into a final context for answering. The default in this study was to inject 15 contextual summaries, but we saw maximal accuracy with 5 at the cost of precision. LLMs were prompted to answer as follows:

> Answer the question below with the context.
> Context: {context} —- Question: {question}
> Write an answer based on the context. If the context provides insufficient information and the question cannot be directly answered, reply "I cannot answer." For each part of your answer, indicate which sources most support it via citation keys at the end of sentences, like (Example2012Example pages 3-4). Only cite from the context and only use the valid keys. Write in the style of a Wikipedia article, with concise sentences and coherent paragraphs. The context comes from a variety of sources and is only a summary, so there may inaccuracies or ambiguities. If quotes are present and relevant, use them in the answer. This answer will go directly onto Wikipedia, so do not add any extraneous information.
> Answer ({answer_length}):

Where contexts are injected by the generate answer code before output is returned to the agent.

Citation Traversal Tool

Atop the PaperQA[4] tools, we created an additional tool to traverse one degree of citations, both forward in time ("future citers") and backwards in time ("past references"). This tool enables a fine-grained search around paper(s) containing relevant information. The traversal originates from any paper containing a highly-scored contextual summary (RCS score 0-10), and our minimum score threshold was eight (inclusive). The papers corresponding to highly-scored summaries are referred to as D_{prev} in **Algorithm 1**. See **Table 1a** for the frequencies of various $|D_{\text{prev}}|$ when this tool was selected.

Table 1: Various statistics on citation traversal.

$\lvert D_{\text{prev}}\rvert$	1	2	3	4	5	6	7	8	9	10
Count	2147	941	530	386	307	216	154	67	29	12
Frequency (%)	44.8	19.6	11.1	8.1	6.4	4.5	3.2	1.4	0.6	0.3

(a) Distribution of traversal starting paper count $|D_{\text{prev}}|$.

$\lvert D_{\text{prev}}\rvert$	1	2	3	4	5	6	7+
Frequency of 1 Overlap (%)	**100.0**	**91.3**	**91.7**	90.8	90.8	88.7	87.5
Frequency of 2 Overlaps (%)		**8.7**	**7.4**	**7.5**	**7.2**	**8.2**	8.5
Frequency of 3 Overlaps (%)			**0.9**	**1.4**	**1.4**	**2.0**	**2.4**
Frequency of 4 Overlaps (%)				**0.3**	**0.5**	**0.7**	**0.9**
Frequency of 5 Overlaps (%)					**0.1**	**0.3**	**0.4**
Frequency of 6 Overlaps (%)						**0.1**	**0.2**
Frequency of 7+ Overlaps (%)							**0.1**

(b) Table showing the frequencies of citation overlap o seen in LitQA, illustrating the percentage of traversed citations at stake when filtering with an overlap threshold θ_o . We chose to specify $\theta_o = \lceil \alpha \times |D_{\text{prev}}| \rceil$, where α is known as the overlap fraction and was defaulted to $\frac{1}{3}$. The bolded values show what overlaps would have been preserved using an $\alpha = \frac{1}{3}$.

To first acquire citations, Semantic Scholar[5] and Crossref[13] APIs are called for past references and Semantic Scholar APIs are called for future citers. To collect all citations for a given paper, we make one API call per provider per direction, totalling four API calls/paper. All three providers only provide partial paper details, meaning a large fraction of the time a title or DOI is not present in the response metadata. To merge citations across providers, a best-effort de-duplication is performed using casefolded title and lowercased DOI. In **Algorithm 1**, this logic takes place inside the `GetCitations` procedure.

Once citations have been acquired, bins of overlap $\mathcal{B}$ are computed. For example, traversing past references for the following six DOIs: `10.1016/j.mcn.2006.08.007`, `10.1002/cpsc.17`, `10.1002/(sici)1098-`

1136(200004)30:2<105::aid-glia1>3.0.co;2-h, 10.1089/scd.2015.0244, 10.1002/glia.22882, and 10.1042/an20120041, leads to one DOI cited by four papers, five DOIs cited by three papers, 29 DOIs cited by two papers, and 428 DOIs cited by just one paper.

To filter bins of overlap, a hyperparameter "overlap fraction" α was introduced to compute a threshold overlap θ_o as a function of the number of source papers ($|D_{\text{prev}}|$). For example, with an $\alpha = \frac{2}{5}$ and traversing from six source DOIs, all citations not appearing in at least three source DOIs were discarded. The default overlap fraction used in data collection was $\frac{1}{3}$. See **Table 1b** for a full distribution of overlaps seen during LitQA runs. Furthermore, a twelve paper limit ℓ was posed on the traversal, which meant in the above example only keeping six of the bin of 29 DOIs cited by two papers. To filter within a bin, we fall back on the count of future citers. This winnowing logic is detailed across **Algorithm 1**'s `FilterOverlap` and `TraverseCitations` procedures. Lastly, we traverse both future citers and past references, feathering together the resultant DOIs before finding them.

Algorithm 1 Traverse Citations

Require: Set of summaries S, score threshold θ_{score}, overlap fraction α, look future flag $\mathbb{1}_{\text{fut}}$, limit ℓ
Ensure: Set of traversed papers D_{out}, where papers are future citers if $\mathbb{1}_{\text{fut}}$ else past references

procedure TRAVERSECITATIONS(S, θ_{score}, α, $\mathbb{1}_{\text{fut}}$, ℓ)
  $D_{\text{prev}} \leftarrow \{s_d \mid s \in S \wedge s_{\text{score}} \geq \theta_{\text{score}}\}$ ▷ Traverse from highly-scored summaries' corresponding papers
  $\mathcal{D} \leftarrow$ GETCITATIONS($D_{\text{prev}}, \mathbb{1}_{\text{fut}}$) ▷ $\mathcal{D}$ is a set of sets of papers such that $|\mathcal{D}| = |D_{\text{prev}}|$
  $\theta_o \leftarrow \lceil \alpha \times |\mathcal{D}| \rceil$ ▷ Overlap threshold θ_o scales with $|S|$
  return FILTEROVERLAP($\mathcal{D}$, D_{prev}, θ_{o}, ℓ)

Require: Set of sets of candidate papers $\mathcal{D}$, set of previous papers D_{prev}, (inclusive) overlap threshold θ_o, limit ℓ
Ensure: Set of filtered papers D_{out}

procedure FILTEROVERLAP($\mathcal{D}$, D_{prev}, θ_{o}, ℓ)
  $\mathcal{B} = \left[\left(o, \{d \mid \left(\sum_{D \in \mathcal{D}} \mathbb{1}(d \in D)\right) = o\}\right) \textbf{ for } o \in [|\mathcal{D}|, \ldots, 1]\right]$ ▷ Bin papers according to decreasing overlap
  $D_{\text{out}} \leftarrow \{\}$
  for $o, D \in \mathcal{B}$ **do** ▷ Highest overlapping citations come first
    if $o < \theta_o \vee |D_{\text{out}}| \geq \ell$ **then break**
    $D \leftarrow \{d \mid d \in D \wedge d \notin D_{\text{prev}}\}$ ▷ Filter out already present papers
    if $\ell - |D_{\text{out}}| < |D|$ **then** ▷ If the entire bin won't fit within limit ℓ
      $D \leftarrow \{d \mid i, d \in \text{SORT}_{\downarrow \text{CITERS}}(D) \wedge i \leq (\ell - |D_{\text{out}}|)\}$ ▷ Keep subset with the most future citers
    $D_{\text{out}} \leftarrow D_{\text{out}} \cup D$
  return D_{out}

8.2 LitQA

8.2.1 Question Construction and Human Evaluation

LitQA questions were generated manually by a combination of the authors as well as contracted human experts (see [14]). All human annotators were compensated, informed that their evaluations were being used in research of human-level performance, and consented to the use of their annotations and participation. Question authors were instructed to identify recent papers (published within the last 36 months), and develop a multiple-choice question that requires context within the main text of the paper to answer and *is not answerable by the abstract or title alone.* They were further advised that the question should require some amount of reasoning within the paper context, and not be a direct quote or statement from the paper. Distractors were instructed to be reasonable within the context, using either other information in the paper (e.g. other genes being discussed) or based on inherent or other knowledge. We also periodically tested question drafts with ChatGPT 3.5 or 4 (with logging disabled) to ensure questions were not easily answerable by models already, or to help design effective distractors by asking to provide plausible answers. Question drafts were also often searched against Google Scholar to ensure that it was not trivial to find the exact statement necessary to answer the question. This was an effective aid in revising question wording.

For contracted question authors or anyone new to drafting, the first 10 or so questions produced were carefully reviewed by one or more of the authors for quality, after which they were either asked to rework them, they were added to corpus, or they were removed from consideration. This feedback (referred to as `calibration`) was usually enough to ensure quality question generation going forward from contractors, though questions were reviewed by authors on an ongoing

basis prior to merging into the main corpus. Contractors were compensated via multiple structures during different phases of generation as we iterated on effective strategies. Initially, they were paid on an hourly basis at $50 per hour. At later phases, they were paid on a completion basis such that they were paid per accepted question, which equated to variable hourly equivalents always $50 or more per hour. Using this methodology, LitQA2 was built up from LitQA (47 questions) in two stages of releases, first 100 questions (147), then an additional 101 questions, adding to the original subset to make 248 total questions.

Human evaluators were assigned questions from the LitQAv2 corpus in rounds of "quizzes" composed of 20-40 questions each. They were allocated up to a week to complete the quiz, but were given no other time constraints. They were also allowed to use tools such as internet search or journal collection search provided via their institutions. They were asked to explicitly refrain from using AI-based tools such as ChatGPT or Claude, though we did not have any method of enforcement of this request. In order to encourage completion as well as high performance, we attempted to design an incentive scheme that separately promoted both: In brief, we paid a base dollar amount (from $3-6) to each question for completion. We then added a bonus to each question that was based on overall performance:

- 80% or more overall score = bonus equal to base question amount for each correct question.
- 60% - 80% overall score = bonus equal to half the base question amount for each correct question.
- Less than 60% overall score = bonus equal to $1 for each correct question.

We additionally paid a completion bonus for the entire quiz in some instances, amounting to $150. On average, evaluators were paid between $50 to $100 per hour based on self-estimation of time spent and total compensation. Quizzes were sometimes combined with similar evaluation quizzes for other evaluation benchmark categories we are developing.

For the human LitQA2 performance reported in this work, 2 rounds of quizzes (20 questions each) were given to 9 evaluators. In total, evaluators answered provided 266 unique answers across 248 unique questions.

A third round of quizzes was given with another set of 160 questions given to evaluators, with questions which overlapped from both the initial set of 147 questions and the remaining 101 questions. However, this quiz round had several outlier scores (>90%) in quizzes on the same subsets of questions which had been given previously to a much lower average (66.3%). Upon investigating, we found that a portion of the initial 147 LitQA2 questions that were available on GitHub had been parsed by google-indexed data aggregators, causing it to trivially return results with a Google search from evaluators. Interestingly, this did not impact PaperQA2 performance, because PaperQA2 does not use Google websearch. Thus, we chose to exclude evaluator answers to the initial 147 LitQA2 questions in the third set of quizzes as it was no longer a valid human comparison.

8.2.2 PaperQA measurement

When measuring LitQA2 performance, `paperqa` function calls were performed for each LitQA question. LitQA answer order was randomized for each call, and, for all configurations shown, 3 full runs of all LitQA questions were performed. 3 runs were necessary to control for the inherent noise in LLM inference, even at 0 temperature (where all tests were performed). LitQA2 was automatically evaluated using an evaluation LLM call (GPT-4-0613), which extracted the letter answer from PaperQA2's output. It was prompted to extract as follows:

> Extract the single letter answer from the following question and answer {prompt_output('QA')}
> Single Letter Answer:

The extraction was then parsed and graded against the LitQA2 benchmarks correct ideal answer. All questions were graded as "Unsure" if the "Insufficient information to answer this question" option was selected, except for the LitQA2 questions where the ideal answer is "null", then the "Insufficient information" was graded as "Correct".

LitQA2 was developed in two stages of 100 then 101 questions, adding to the original subset of LitQA (47) questions to make 248 questions. This gave us an opportunity to evaluate our system before and after adding the set of 101 previously unseen questions to evaluate overfitting. We can see that our system performed similarly in terms of accuracy between question sets, implying we were not overfitting to the original LitQA data.

When comparing PaperQA2 to other RAG technologies in **Figure 2A**, the author's manually entered each LitQA2 question (in the exact format given to PaperQA2) into the online interfaces for both Perplexity [15] and Elicit [16]. Perplexity was run using the "Pro" product, and GPT-4o, across all 248 LitQA2 questions. We recorded either the letter answer, or manually interpreted the response to assign the system's letter output. The performance metrics were calculated in an identical way to PaperQA. Elicit was not able to be run for all 248 questions, the numbers reported only represent the first tranche of 147 LitQA2 questions. On Elicit's final 101 LitQA2 questions, the authors noted a significant algorithm

Ablation	Accuracy (Old 147)	Accuracy (New 101)	Precision (Old 147)	Precision (New 101)
Claude-3-Opus	59.0% ±2.4%	54.5% ±6.1%	93.9% ±3.7%	82.6% ±6.0%
Gemini-1.5-Pro	55.8% ±5.7%	59.4% ±2.3%	93.2% ±1.8%	83.0% ±4.6%
GPT-4-Turbo	64.4% ±1.8%	63.7% ±7.4%	91.3% ±2.1%	81.8% ±7.3%

Table 2: Table of model choices using both stages of LitQA2 development, all errors shown are CI intervals.

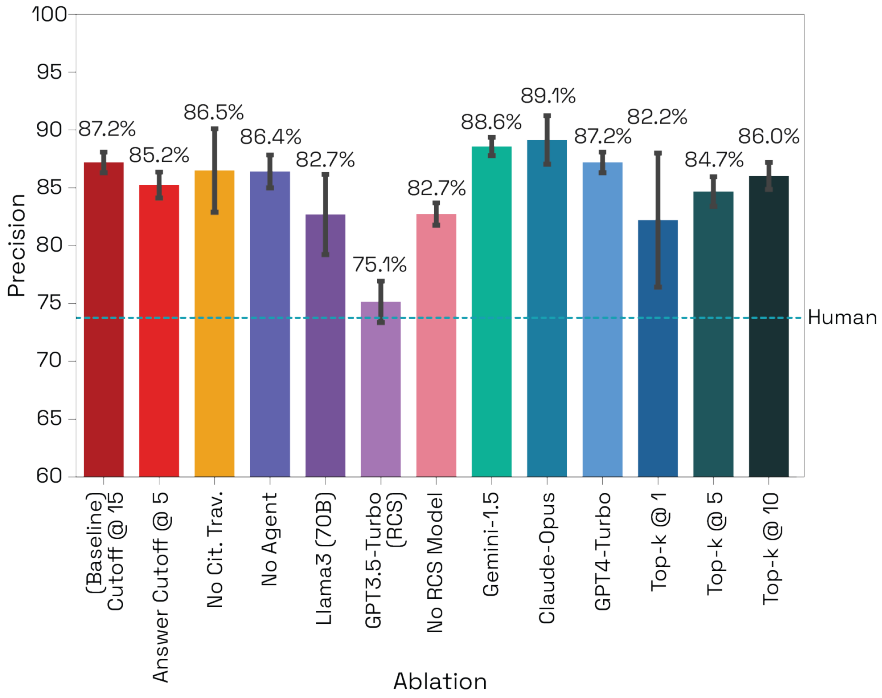


Figure 5: PaperQA2 precision performance on LitQA2 across all configuration categories included in Figure 2. All error bars are 95% CI intervals.

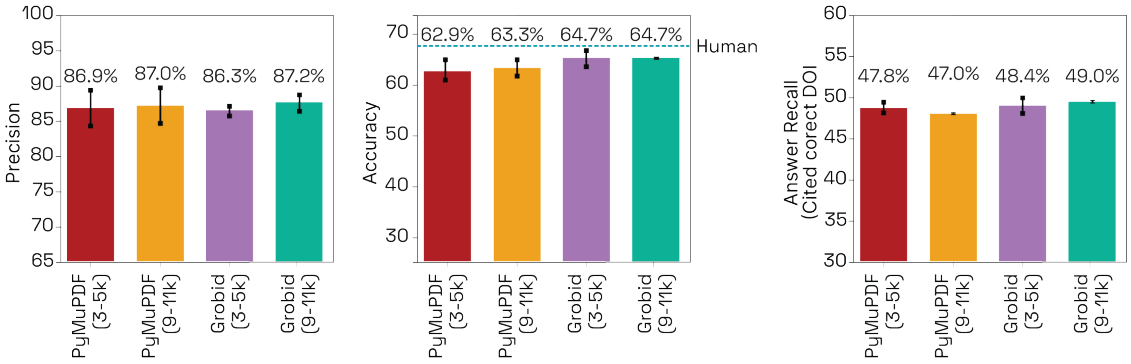


Figure 6: Precision, accuracy and recall for different chunk size and parsing algorithm choices on LitQA.

change. Answers resulted in > 90 % "unsure" answers, thus we are only reporting on the subset of LitQA2 where multiple choice answers could be extracted. We also compared PaperQA2 to PaperQA's prior published configuration[4], which used a smaller RCS model (GPT 3.5 Turbo) and ranking depth, had no citation traversal tool, and did not have access to Grobid parsings. PaperQA's LitQA2 accuracy was 36.7% and its precision was 76.5%. Thus, PaperQA2 showed a large improvement in both metrics, particularly in accuracy.

8.3 WikiCrow

WikiCrow statements were generated using a linked set of 5 different queries, 4 to PaperQA2, and 1 to a frontier LLM model, GPT-4-Turbo. As detailed in **Figure 3A**, each PaperQA2 query created a different article section. The PaperQA2 prompts and settings used for these queries can be found in our `paperqa` repository. The primary PaperQA2 parameter differentiation between WikiCrow and LitQA2 runs was the usage of Grobid for document parsing, as well as the modification of the RCS step to extract a gene name that is the focus of each paper chunk being summarized. They were designed to reduce hallucinations and accurately read data from tables.

We compared WikiCrow and Wikipedia statements starting from a sample of 300 genes which were sampled from from a complete list of all protein encoding human genes[17] (n=19,255) after being filtered for only those having valid (non-stub) Wikipedia articles (n=3,639). Among the 300 articles, Wikipedia statements were sampled, first from the pool of 300 genes, then among each article's <p> HTML sections with more than 25 characters. 500 total Wikipedia statements were sampled, which resulted in 240 unique genes. Matching WikiCrow articles were generated for each of the 240 genes for comparison. WikiCrow statements were broken at either a reference list (notated with parenthesis and PaperQA2 document names) or at a paragraph break, most closely matching Wikipedia's HTML structure. Each of the 500 Wikipedia statements were randomly matched with a WikiCrow statement from the same gene article to ensure that evaluators were given a similar gene-distribution of statements between WikiCrow and Wikipedia articles.

References were extracted from the Wikipedia statements using the <sup> link, and matched to the correct DOI or hyperlink in the article references. References were extracted from the WikiCrow statements by parsing the parenthetical insertions (via regular expressions) and matching the DOI to the document name given by the model. In this way, each statement was associated with a list of DOI links to the sources listed. Efforts were then made to structure each statement similarly, removing any HTML from the Wikipedia statements, and removing all inserted references into the WikiCrow statements. All references were replaced with random numbers between 1 and 30, but kept consistent if listed several times in the same statement. References were then uniformly reformatted as `[x]`, to hide their origin.

The matching 500 Wikipedia and 500 WikiCrow statements were shuffled for each of the 4 evaluators. Each was given a list of 200 to grade while blinded to the statement origins. Graders were sent the questions with the following instructions:

> **Task overview**
>
> You'll be reviewing statements written about human genes. These statements are intended to provide information taken from the literature, and should thus provide an accurate citation such that the stated information matches information contained in the cited literature. You'll be evaluating each statement according to its accuracy as well as being cited appropriately. Please read the guidelines below carefully to ensure accurate work.
>
> **Scoring**
>
> You'll soon receive a link to a private Google Sheet with 200 statements, along with some associated information for each one, including links to any cited literature. You'll be grading the sentences in each statement according to the below metrics. Some statements may contain multiple sentences, each of which should be evaluated such that any sentence not satisfying our scoring criteria counts against the entire statement.
>
> - Is it cited? In other words, does the statement contain one or more citations to published literature?
> - Is the information correct, as cited? In other words, is the information stated in the sentence correct according to the literature that it cites?
>
> There will be a column for each of the above scores. Put TRUE in the appropriate column if true according to the metric, and FALSE if not. There will also be a column for "notes". Please use this column to include anything you think is important for your grading, or other flags we might want to know about. Below are some more detailed guidelines for scoring:
> You are only evaluating if the statements are correct as cited – not if the statement is globally correct or supported by other sources. Some common scenarios you may encounter:
>
> - **Simple statements:** Statement has a single citation and the content is supported by the citation. Is Cited = TRUE, Is Info Correct As Cited = TRUE

- **Simple unsupported statements:** A statement is true, highly-specific (see "broad context" below, but not explicitly stated in the source, mark as FALSE for “correct as cited”, but mark "is cited" as TRUE.
- **Unrelated, meta statements:** Statement is a meta-statement or unrelated to biology. For example, the statement may be the start of a list like: "Gene XYZ has the following properties:", or an attribution like "This data has been provided by Y organization." These statements should be judged as "Not Applicable/NA". They do not need to be cited.
- **Broad context:** Statement has a citation and the content is a broader context than the citation, but not explicitly explained in the citation. For example, an explanation of what a secondary structure is or what a protein sequence is. The broad content should be judged to be "undergraduate biology student common knowledge". If it is, then the info can be marked correct as cited, if it’s more advanced than undergraduate bio knowledge (or outright incorrect), then the info is incorrect as cited.
- **Underspecified citation:** The statement may be factually correct, but the citation is under-specified. An example might be that the citation is a link to a database, where a user needs to query the database and synthesize their own data to support the statement. If it requires synthesis by the user, than the statement should be judged as NOT correct as cited. If it’s a database link to the a page about the gene, and the fact can be found there, then it can be judges as correct as cited.
- **Inaccessible citation:** The statement may be factually correct, but the citation is to an inaccessible source, like a closed source database or a textbook. These should be judged as "Not Applicable/NA". Though an attempt should be made to acquire the source to validate the information.
- **Multi-part citations:** The statement has several sentences and citations. Each sentence should be correct as cited given all of the previous criteria (including the "Broad contexts" category).
- Occasionally, it may not be clear from context which sentence applies to a source, there may be three sentences in a row with two sources trailing the sentences. In this case if the sentences are supported by either source, then it’s correct as cited. If a single sentence is un-supported by either source, and they don’t meet our broad context criteria, then mark as incorrect as cited.
- If there are multiple citations, where one is inaccessible, but the accessible citations can support the statements by themselves, then mark as correct as cited. If the statement can NOT be validated by the accessible sources, then mark as "Not Applicable/NA".

Evaluators were incentivized to encourage both quality of review and quantity of statements reviewed. They were offered $10 per statement graded as a base payment, and were additionally offered a $10 per question bonus payment if they reviewed more than 50 questions by a specific deadline approximately five days after starting.

Using this criteria, 4 expert researchers were contracted as evaluators to grade 375 evaluations. They were instructed to only complete their statements in the order provided, and any statements that were not graded in a contiguous block from the initial statement were excluded from analysis (to avoid biasing towards simple statements). Among our 375 evaluations, 40 statements overlapped between 2 or more evaluators. Among those 40, 77.5% (31) had agreement, and 22.5% (9) had a disagreement in evaluation, showing good overall alignment in scoring.

We had previously reported[18] lower “cited and unsupported” percentages for both Wikipedia and WikiCrow, but the prior study differed in key ways: 1. this work limits the statement samples to be from the same subset of gene articles, eliminating selection bias for longer, higher quality articles; 2. this work uses statements built from sentences which may cite one or more sources as is typically present in articles, the prior work focused on only single sentences with one source which greatly limited the sample; and 3. this work used a much more robust evaluation pipeline, with blinded external evaluators.

Due to the complexity and time required to evaluate WikiCrow accuracy, we relied on heuristics to evaluate our summarization-centric features, adding metadata (gene name) extraction to our RCS step and Grobid-based parsings. These features were meant to mitigate gene name confusion between the RCS step and the generate answer tool, and misreading tabular data, which anecdotally were the two most frequent issues found by the authors while developing WikiCrow. Among the reasoning issues reported by the evaluators, only 2 / 171 statements had issues with gene name confusion, and there were no reported table extraction errors. Utilizing Grobid structured parsings also reduced PaperQA2’s token usage. Across a sample of 5,363 papers, PyMuPDF’s parsings resulted in an average of 16,040 tokens per paper, while Grobid’s parsings resulted in an average 8,903 tokens. The savings can be attribution to a reduction in excess whitespace as well as the removal of reference sections, which we did not include in our Grobid parsings.

8.4 Contradiction Detection Methods

ContraCrow detects contradictions in literature through two steps. Unless provided, the first step in ContraCrow is to extract claims from a given paper **Figure 4A**. First, the provided paper is split into chunks. To maintain context for extracting claims, each chunk is only split within its respective section, up to 5000 characters (no overlap), and the section and paper titles are included. Each resulting chunk is then fed into a claim-extraction LLM to extract claims candidates. The resulting extracted claims are each evaluated by a filtering LLM that assesses the claims based on quality and generalizability, assigning a score out of 10. Only claims scoring 8 or higher are considered.

Once claims have been generated or are provided, each claim is fed into PaperQA2 with a special *contradiction detection prompt*, instructing the system to search for contradictions and respond with an appropriate format. The system considers each claim independently and identifies relevant papers across literature in order to identify possible contradictions to the claim. Finally, as instructed by the *contradiction detection prompt*, ContraCrow outputs its reasoning and a choice from an 11-point Likert scale (see Figure 4D). We map the natural language score to integers (0-10) in our calculations, which allows us to tune the decision threshold, as seen in 4B. For all experiments, this step uses Claude 3.5 Sonnet[19] as the LLM model, a chunk size of 7000 characters, a temperature of 0, and a simple overlap of 250. Unless otherwise specified, we use a decision boundary of 8 on this step.

8.4.1 ContraDetect

The ContraDetect benchmark was generated from the LitQA2 contradiction dataset, removing any questions with "null" responses as to not overlap with the no-evidence data. The resulting questions were randomly split into two groups. The questions and corresponding ideal answers from the first group were individually given to `gpt-4-turbo-2024-09`[20] with instructions to rephrase them into factual statements. The second group was similarly turned into incorrect statements (contradictions), given question and the first corresponding distractor answers. Each question corresponds to exactly 1 statement. These claims then proceeded into ContraCrow's second step, as described above.

We separately designed 42 no-evidence statements by hand. The authors carefully designed questions in their respective fields of expertise and did extensive literature review to ensure that no literature has reported on the claim. There is no overlap between the no-evidence statements and the LitQA2 questions. The resulting answers from ContraCrow were also evaluated by the authors to ensure that no evidence had been found. Any claims with found evidences were removed from this dataset.

8.4.2 Human Evaluation

We randomly selected 100 papers from our local database of biology papers. 7 of these papers failed to parse, leaving our dataset at 93 papers. These papers have no overlap with the papers corresponding to LitQA2 questions. `gpt-4-turbo-2024-04-09`[20] was used for both LLM models in the claim-generation step. The resulting claims were then fed into the contradiction-detection step as described above, in batches of 1,000 claims.

For the human "contradiction validation" task, 50 claims scored at 8 and 50 claims at least 9 by ContraCrow were randomly chosen (distribution: $8:50, 9:48, 10:2$). The resulting 100 contradictions were split over 5 expert evaluators evenly, with no overlap. The evaluators were provided with the claim, the original claim information (source paper reference and the chunk used to generate the claim), some background information (acronyms, definitions, etc.), ContraCrow's reasoning, and all chunks cited in the reasoning. The evaluators were tasked with giving two separate binary responses indicating their agreement with both the model's reasoning and conclusion, based on the provided cited chunks (up to 10). They were instructed to use additional provided resources sparingly, only for clarification, and to *only* consider the evidence chunks and ContraCrow's reasoning when making a final decision. Only the label for "agreement with the model's conclusion" was used for analysis.

For the human "contradiction detection" task, 10 claims scored 0-4, 10 claims scored 8, and 10 claims scored at least 9 by ContraCrow were randomly chosen (distribution: $0:1, 1:6, 2:2, 3:1, 8:10, 9:10$). The resulting 30 claims were split over 5 evaluators. Each claim was evaluated by at least 2 evaluators. The evaluators were provided with the claim, the original claim information (source paper reference and the chunk used to generate the claim), background information (acronyms, definitions, etc.), and all literature chunks considered by ContraCrow (up to 15). The evaluators were tasked with assigning a Likert scale value and a binary ("yes" or "no") label to determine whether each claim contained a contradiction to the provided sources. For this task, the evaluators did not have access to ContraCrow's reasoning. Similar to the first task, they were instructed to use any provided materials for clarification, but to only use the provided literature when making a final decision.

In the both the "contradiction validation" task and the binary "contradiction detection" task, ContraCrow scores of $>= 8$ were considered the positive case in order to compare to humans' binary outputs. For the Likert score tasks, raw scores were used.

Evaluators were compensated for their work on grading claims to encourage thorough review. For the "contradiction validation" task, they were paid $12 per claim completed. For "contradiction validation", they were paid $20 per claim completed. They were additionally paid a $200 'completion bonus' for completing grading on all claims by a deadline approximately one week after starting.

Evaluators were sent the questions with the following instructions:

Contradiction Detection Instructions
Task overview
For this task, we are interested in identifying and rating contradicting research findings, or claims. We've assembled a set of claims from research papers that are potentially contradicted by other findings, either prior or later. Your task will be to both assess the claim itself, as well as review portions of text taken from research papers that may be related to and may or may not present contradicting information to the presented claim. We will not have an explicit calibration phase for this task, nor do we provide examples. More detailed instructions are provided below to help guide you. You may use any sources available to you in order to understand concepts, acronyms, etc. that you are unfamiliar with, but you should perform the evaluation itself by-hand and using only the claim and context provided. We also explicitly ask that you do not use ChatGPT or other AI tools to come to your conclusions, though they may be used if helpful to define terms or clarify concepts. Please do not put claims or chunks into ChatGPT and similar tools.

Instructions
This task will consist of two separate activities, Claim Evaluation and Contradiction Detection (type a and b). For each claim, you will be given the following information:

- Claim: a single sentence making some claim or finding.
- Chunk: the section of text from the source paper that was used to generate the claim.
- Section: the section of the paper that the chunk lives in, if available
- Title: the title of the paper that the chunk lives in, if available
- DOI: the DOI URL for the paper that the chunk lives in, if available

Section 1: Claim Evaluation
Your first task is to a determination of the claim being a valid claim. Some examples of good & bad claims are shown below:

- Good claims are typically clear, testable, and generalizable
  - "Cells expressing ARG1 are more likely to undergo apoptosis."
- Bad claims will come in different forms, like not having enough context, i.e. it does not make sense outside the context of the paper.
  - "Cells in cluster 1 had higher expression of ARG1."
  - "Cells expressing this gene are more likely to undergo apoptosis."
  - "Method A performed better than Method B in our study."
  - Claims provided may also just not really be defined as a 'claim'. These might describe methods, descriptions, or obvious common knowledge
    - "We used method A to test the effects of ARG1 on cell death."
    - "ARG1 is a protein."
  - Claims may also not be supported by the provided chunk (Mixed up gene names, off-topic discussion, etc.)

To evaluate each claim, select all that apply from the following options:

- good claim
- not enough context
- not generalizable
- not a claim
- not supported

Some important considerations:

- Multiple claims may come from the same chunk & claims may be redundant in nature
- Consider each claim independently
- You may use the all provided information in this section to help you on this step (including the paper title, as it may provide context)
- Avoid:
  - Evaluating the claim for truth value
  - Evaluating the wording/syntax of the claim, unless it changes the meaning
  - Deep-diving into the paper outside of the claim
  - Spending more than a few minutes grading each claim, on average

Section 2: Contradiction Detection
This part will contain additional information for each claim, consisting of up to 15 "evidence chunks" which are sections of research paper text that may serve as evidence either supporting, contradicting or neutral to the claim to various degrees. Each chunk will itself contain (when available) the source paper's citation. This information can be used for reference and clarity (for defining acronyms, clarifying things, etc) but should not be considered when determining contradictions. Your job for this part of the task is to determine whether (and to what extent) the claim contradicts or agrees with the provided evidence chunks. You will provide a selection for both of the below categories (note there are two separate selections!)

- Contradiction grade (choose one of the following):
  - Explicit Agreement
  - Strong Agreement
  - Agreement
  - Possibly an Agreement
  - Lack of Evidence
  - Possibly a Contradiction
  - Nuanced Contradiction
  - Contradiction
  - Strong Contradiction
  - Explicit Contradiction
- Contradiction determination:
  - YES - Claim contradicts the provided context
  - NO - Claim does not contradict the provided context

Some important considerations:

- A claim may have multiple chunks from the same paper.
- Consider claims individually. If the claim is a bad claim, use your best judgement about whether or not and how to evaluate the contradiction. We ask that you attempt to evaluate it if possible, and if you feel you cannot proceed, please explain why then move on to the next claim.
- Focus on the claim + context provided: Don't worry about other contradictions or agreements in the context, only focus on the specific claim. Additional information (publication information, background info) is provided for clarification where needed, but you should rely solely on the provided context and claim for labeling.
- It is possible that the provided statement makes more than 1 claim. Use your best judgement in this case, considering the context provided.
- Avoid:
  - Evaluating the claim or evidences for truth value or quality (only evaluate in relation to each other).
  - Deep-diving into any text outside of the provided context (including the papers from which the context was derived).

Contradiction Validation Instructions
Task overview
This task is very similar, but this time you will be given an AI model's reasoning on the evidence chunks, and only the evidence chunks the model considered (which should cut down the review time considerably). You will do the usual annotation, and additionally say whether you agree with the LLM's reasoning. Section 1 is unchanged, Section 2 is different.

We also explicitly ask that you do not use ChatGPT or other AI tools to come to your conclusions, though they may be used if helpful to define terms or clarify concepts. Please do not put claims or chunks into ChatGPT and similar tools.

Instructions

This task will consist of two separate activities, Claim Evaluation and Contradiction Detection (type a and b). For each claim, you will be given the following information:

- Claim: a single sentence making some claim or finding.
- Chunk: the section of text from the source paper that was used to generate the claim.
- Section: the section of the paper that the chunk lives in, if available
- Title: the title of the paper that the chunk lives in, if available
- DOI: the DOI URL for the paper that the chunk lives in, if available

Section 1: Claim Evaluation

Section 1 is the same as in the "contradiction detection" task.

Section 2: Contradiction Detection - Model Evaluation

This part will contain additional information for each claim. Each claim will have a model response, which is some reasoning to answer the question 'Are there any contradictions to this claim in literature?' The model response will reference up to 10 "evidence chunks" which are sections of research paper text. These referenced chunks will also be provided for you and can be linked via the citation key in the model response (for example, the model response might reference `Song2022DEPTH2:pages 16-17`. The chunk corresponding to this reference will also be found under `Song2022DEPTH2:pages 16-17`. Each chunk will itself contain (when available) the source paper's citation. This information can be used for reference and clarity (for defining acronyms, clarifying things, etc) but should not be considered when evaluating contradictions or model response. Your job for this task is to determine whether you agree with the model response and to what extent the claim contradicts or agrees with the evidence chunks provided. You will provide a selection for both of the below categories (note there are two separate selections!)

- Contradiction grade (choose one of the following):
  - Explicit Agreement
  - Strong Agreement
  - Agreement
  - Possibly an Agreement
  - Lack of Evidence
  - Possibly a Contradiction
  - Nuanced Contradiction
  - Contradiction
  - Strong Contradiction
  - Explicit Contradiction
- Contradiction determination:
  - YES - Claim contradicts the provided context
  - NO - Claim does not contradict the provided context
- Optional Explanation: optionally, you may provide a written explanation/justification of your scoring.

Some important considerations:

- A claim may have multiple chunks from the same paper.
- Consider claims individually.
- Because the model response is provided, we ask that you evaluate every provided claim.
- Additional information (publication information, background info) is provided for clarification where needed.
- It is possible that the provided statement makes more than one claim. Use your best judgment in this case, considering the model response provided.
- Avoid:
  - Evaluating the claim or evidences for truth value or quality (only evaluate in relation to each other).

- Evaluating the model response truth value, quality, syntax, or style, except where directly related to the claim.
- Deep-diving into any text outside of the provided context (including the papers from which the context was derived).

References

[1] Harrison Chase. Lanchain. `https://github.com/langchain-ai/langchain`, 2022.

[2] Dagster. `https://docs.dagster.io/`.

[3] Kubernetes. `https://kubernetes.io/`.

[4] Jakub Lála, Odhran O'Donoghue, Aleksandar Shtedritski, Sam Cox, Samuel G Rodriques, and Andrew D White. Paperqa: Retrieval-augmented generative agent for scientific research. *arXiv preprint arXiv:2312.07559*, 2023.

[5] Rodney Kinney, Chloe Anastasiades, Russell Authur, Iz Beltagy, Jonathan Bragg, Alexandra Buraczynski, Isabel Cachola, Stefan Candra, Yoganand Chandrasekhar, Arman Cohan, et al. The semantic scholar open data platform. *arXiv preprint arXiv:2301.10140*, 2023.

[6] Tiktoken. `https://github.com/openai/tiktoken`.

[7] Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. Query rewriting for retrieval-augmented large language models. *arXiv preprint arXiv:2305.14283*, 2023.

[8] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2023.

[9] Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D Manning. Raptor: Recursive abstractive processing for tree-organized retrieval. *arXiv preprint arXiv:2401.18059*, 2024.

[10] Shengyao Zhuang, Bing Liu, Bevan Koopman, and Guido Zuccon. Open-source large language models are strong zero-shot query likelihood models for document ranking. *arXiv preprint arXiv:2310.13243*, 2023.

[11] Yunfan Gao, Tao Sheng, Youlin Xiang, Yun Xiong, Haofen Wang, and Jiawei Zhang. Chat-rec: Towards interactive and explainable llms-augmented recommender system. *arXiv preprint arXiv:2303.14524*, 2023.

[12] Yue Yu, Wei Ping, Zihan Liu, Boxin Wang, Jiaxuan You, Chao Zhang, Mohammad Shoeybi, and Bryan Catanzaro. Rankrag: Unifying context ranking with retrieval-augmented generation in llms. *arXiv e-prints*, pages arXiv–2407, 2024.

[13] Crossref. `https://www.crossref.org/`. Accessed: 2024-07-10.

[14] Jon M. Laurent, Joseph D. Janizek, Michael Ruzo, Michaela M. Hinks, Michael J. Hammerling, Siddharth Narayanan, Manvitha Ponnapati, Andrew D. White, and Samuel G. Rodriques. Lab-bench: Measuring capabilities of language models for biology research. *arXiv preprint arXiv:2407.10362*, 2024.

[15] Perplexity. `https://www.perplexity.ai/`. Accessed: 2024-07-10.

[16] Elicit. `https://www.elicit.com/`. Accessed: 2024-07-10.

[17] Wikipedia contributors. List of human protein-coding genes 1 — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/w/index.php?title=List_of_human_protein-coding_genes_1&oldid=1221111978`, 2024. [Online; accessed 11-July-2024].

[18] Sam Cox, Michael Hammerling, Jakub Lála, Jon Laurent, Sam Rodriques, Matt Rubashkin, and Andrew White. Wikicrow: Automating synthesis of human scientific knowledge. `https://www.futurehouse.org/wikicrow`.

[19] Introducing the next generation of Claude — anthropic.com. `https://www.anthropic.com/news/claude-3-family`. [Accessed 13-07-2024].

[20] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.