AnalystBench: Benchmarking professional long-form report generation with web-mined multimodal tasks

Chau Minh Pham[1] **Zichao Wang**[2] **Puneet Mathur**[2] **Alexa Siu**[2] **Akriti Jain**[2]
Aparna Garimella[2] **Ananya Sai**[2] **Nedim Lipka**[2] **Mohit Iyyer** [1] **Varun Manjunatha**[2]

[1] University of Maryland, College Park [2] Adobe Research

chau@umd.edu vmanjuna@adobe.com

Abstract

Large language models are increasingly used to draft long-form multimodal documents, but their end-to-end performance on professional report generation remains systematically understudied. We introduce AnalystBench, a *continually extensible* benchmark of 20 real-world report generation tasks grounded in multimodal document collections, where models must process millions of input tokens to produce long-form professional reports. Using expert-validated quality checklists and groundedness evaluation, we evaluate LLMs and coding agents and find that the best-performing model, GPT-5.1, scores highly on executive summary tasks (exceeding 90% on quality checklists) but degrades substantially on tasks requiring long-horizon synthesis over large inputs (down to 25-41%). Agent-based generation substantially benefits strong closed-source models like GPT-5.1, with checklist scores improving by 21.27 percentage points and visual coverage by 39 points over vanilla generation, but provides little benefit, and sometimes negative gains, for open-source models. like DeepSeek-R1 (-2.57 points). Expert reviewers note that while generated reports are grounded and clearly separate factual description from interpretation, they often fall short in actionability and precision, which highlights the remaining gap between system performance and professional needs.

1 Introduction

A central question for LLM evaluation is whether models truly augment human knowledge work, rather than simply displacing labor (Autor, 2015; Shneiderman, 2020). Addressing this question requires evaluations that reflect real-world professional workflows (Brynjolfsson et al., 2023; Dell'Acqua et al., 2023), and a particularly useful testbed is end-to-end *professional report generation*, where analysts synthesize large information sources into long-form documents that follow standardized structures (e.g., regulatory filings, policy briefs, economic outlooks). Existing benchmarks, however, typically decompose this workflow into individual components, including multimodal summarization (Nallapati et al., 2016; Fabbri et al., 2021; Kantharaj et al., 2022; Tang et al., 2023), structured data aggregation (Parikh et al., 2020; Liu et al., 2022), or document generation (Wiseman et al., 2017; Puduppully et al., 2019; Malaviya et al., 2025). While benchmarks like GDPval (Patwardhan et al., 2025) do focus on realistic professional tasks, they rely heavily on expert-designed tasks and manual evaluation, which makes it expensive and slow to scale. Therefore, it remains unclear how to systematically and sustainably evaluate LLMs on multimodal report generation tasks.

We introduce AnalystBench, a benchmark of 20 end-to-end report generation tasks built from publicly available professional documents and validated by human experts. Each task comes in five variants (e.g., different years/categories) and includes (1) a reference report, (2) offline source documents,[1] and (3) a task description reconstructed from the reference report. AnalystBench is designed to stress-test long-form report generation from massive input collections based on three principles: (1) reflecting **realistic** professional workflows; (2) supporting **evergreen** variants across years or categories to scale easily via an automated tool that generates new tasks from arbitrary documents;[2] and (3) requiring **large-scale multimodal** processing, where models have to analyze over 32M BPE tokens[3] on average to produce a report (see an example task in subsection A.2).

Using AnalystBench, we evaluate LLMs and

[1] We disable web search and provide all source documents locally to make sure that the model cannot directly retrieve the reference outputs from the Internet.

[2] It costs $258.32 to construct 20 tasks in the benchmark.

[3] Calculated with tiktoken using o200k_base.

Findings of the Association for Computational Linguistics: ACL 2026, pages 23894–23926

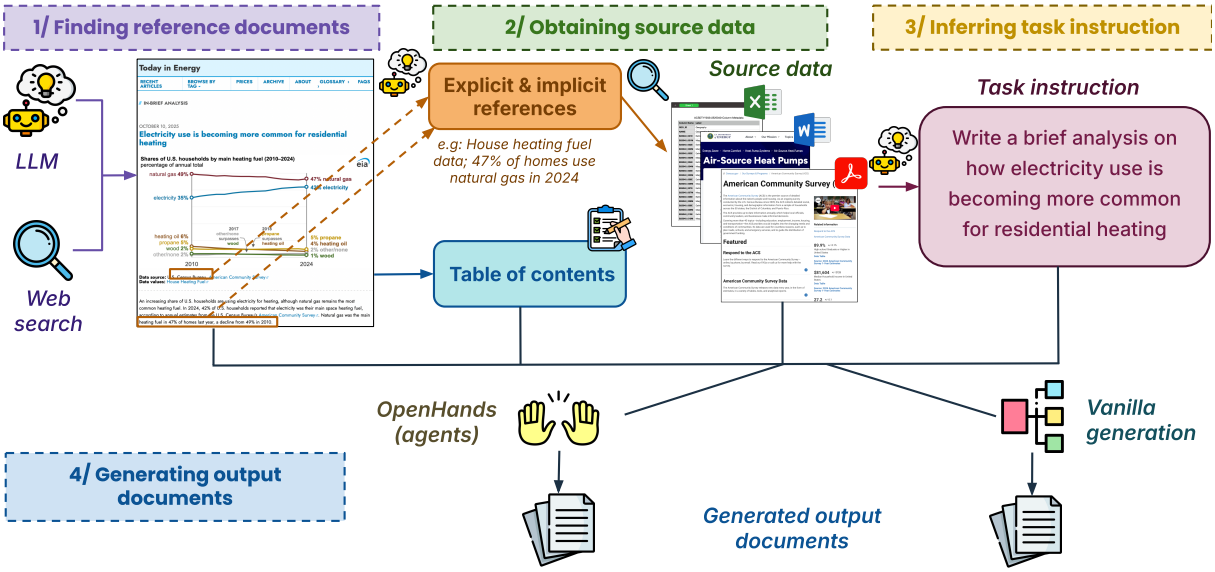

Figure 1: Overview of the AnalystBench construction pipeline. We use LLM prompting and programmatic web search to identify tasks with reference documents $\mathcal{R}$. An LLM then extracts *explicit* and *implicit* source references, which are retrieved as $\mathcal{S}$ via web scraping. Finally, given $\mathcal{R}$ and $\mathcal{S}$, an LLM reverse engineers the task instruction.

coding agents on end-to-end report generation with quality checklists and groundedness metrics. Since long-form generation is difficult to evaluate reliably (Xu et al., 2023), we adopt checklist-based evaluation as a grounded, scalable measure of report quality (Wadhwa et al., 2025; Lee et al., 2025), and validate the checklists with human experts. Overall, GPT-5.1 with OpenHands implementation (Wang et al., 2025) scores 79.51% on quality checklists. Performance exceeds 90% on executive summary tasks with moderate input sizes, but drops sharply to 25-41% on tasks that require long-horizon synthesis across massive inputs. Agent-based generation substantially benefits strong closed-source models such as GPT-5.1 and Gemini-3-Pro, with checklist scores improving by up to 21.27 percentage points and visual coverage by 39 points over vanilla generations. However, this approach offers little or negative benefit for open-source models, as it reduces checklist scores by 2.57 and 4.56 points for DeepSeek-R1 and Qwen3-32B. Providing structural guidance via a template document or table of contents results in larger gains than scaling model size, improving checklist scores by 30.82 points and reducing hallucination rates by 4.64 points with the template document. Finally, qualitative analysis shows that while generated reports are generally structured and separate factual content from speculation, they often lack quantitative precision, presentation quality, and actionability, which limits their readiness for professional use. In summary, our contributions are:

1. AnalystBench, a benchmark of 20 realistic report writing tasks derived from web documents, with tools for creating new variants.
2. A systematic evaluation of LLMs and coding agents on AnalystBench across generation strategies, input settings, and model scales.
3. A taxonomy of failure modes in LLM reports that highlight gaps between generated reports and human writing.

2 Constructing AnalystBench

Each instance in AnalystBench has (1) a task instruction ($\mathcal{T}$), which defines the task goal and output expectations; (2) source documents ($\mathcal{S}$), a collection of materials with all relevant information; and (3) a reference output ($\mathcal{R}$), which provides concrete examples of completed tasks without enforcing a fixed output format. Below, we describe how we construct 100 benchmark instances, which span 20 tasks with five variants each.

2.1 Desiderata

We design AnalystBench around three desiderata that reflect realistic report writing workflows:

➤ **Multimodal processing:** Report writing often requires navigating large collections of files (e.g., long text documents, structured data tables, charts) and synthesizing them into coherent and informa-

Task	Domain	Description	$\mathcal{S}_{\texttt{full}}$	$\mathcal{R}_{\texttt{full}}$	$\mathcal{S}_{\texttt{text}}$	$\mathcal{R}_{\texttt{text}}$
`acts`	Legislative	Summarize new state environmental legislation	2,093,367	30,740	867,189	7,598
`bjp`	Statistical	Summarize criminal victimization from BJS data	97,416	21,619	19,126	756
`bls`	Economic	Summarize employment situation using BLS indicators	201,100	2,562	20,198	2,497
`columbus`	Economic	Compile the city's annual Popular Annual Financial Report	8,008,535	947,655	844,888	14,304
`congress`	Legislative	Generate summaries of U.S. appellate court decisions	478,713	7,371	152,023	1,690
`crash`	Statistical	Summarize crash data on pedestrian fatalities	602,762,580	114,726	3,161	22,738
`delaware`	Legislative	Summarize Delaware M&A corporate law updates	2,687,827	3,501	627,344	3,501
`desa`	Statistical	Write policy briefs with sustainable development factors	3,745,501	14,874	257,707	4,283
`driver`	Policies	Summarize state's driver's license policies for immigrants	1,564,791	3,833	347,871	3,833
`energy`	Economic	Generate EIA analyses of emissions and energy use	10,552,085	66,856	71,398	825
`fed`	Economic	Summarize reports on U.S. household economic well-being	510,558	5,298	46,946	1,657
`fred`	Economic	Produce FRED-style blog posts analyzing economic trends	456,195	45,499	17,878	716
`gao`	Policies	Summarize GAO reports for federal agency CIOs	7,961,602	3,234	528,333	3,234
`kff`	Policies	Write KFF health policy analyses using survey data	702,840	452,204	24,440	4,455
`medicare`	Policies	Produce yearly Medicare Trustees Report fact sheets	526,803	1,719	118,482	1,719
`oecd`	Economic	Write revenue statistics reports for individual countries	1,201,523	47,394	60,960	1,835
`pew`	Statistical	Compile Pew Research fact sheets on media consumption	376,399	72,890	15,550	1,876
`regents`	Legislative	Summarize proceedings of the NY Board of Regents audit	1,902,673	56,610	245,304	7,105
`sandiego`	Statistical	Compile the city's capital improvement budget summaries	5,109,514	109,323	275,118	3,255
`uscis`	Policies	Write policy alerts on updates in immigration procedures	358,260	2,298	160,059	2,298
Average			**32,564,914**	**100,510**	**235,199**	**4,509**

Table 1: Overview of tasks in `AnalystBench`. $\mathcal{S}$ and $\mathcal{R}$ denote BPE token counts for source and reference outputs. Subscripts `full` include figures (base64) and tables, while `text` excludes multimodal elements. Row colors indicate task variants: Yearly (one per year), Categorical (one per category/topic). Full sources are in Table 5.

tive reports. To reflect this setting, our benchmark includes tasks that require models to process mixed-format input collections (e.g. `.md`, `.csv`, `.xlsx`) and generate multimodal reports that contain text, structured data, and visualizations.

➤ **Realistic professional workflows:** We focus on tasks that can be completed using standard productivity tools (e.g., Microsoft Office), which include both summarization- and synthesis-oriented tasks. We also prioritize publicly available documents so that tasks reflect real professional practice and the benchmark can be fully open-sourced.

➤ **Evergreen variants:** A common issue in existing benchmarks is contamination, especially when models are trained on public datasets (Brown et al., 2020; Dodge et al., 2021; Sainz et al., 2023; Deng et al., 2024). We therefore select tasks that allow additional instances over time (e.g., annual updates, entity-specific variations). We formalize them as *variants* $\mathcal{V}$, which are distinct instantiations of the same underlying task. Each task includes five such variants, and we provide a tool that helps automatically refresh the dataset to maintain relevance and measurement validity over time.

2.2 Reverse-engineering tasks from the web

We construct `AnalystBench` using a reverse engineering approach: starting from a real professional report collected from the web, we identify the underlying source materials and derive the corresponding task instruction. This strategy is necessary because high-quality datasets that pair realistic source documents with long-form output reports are rare and hard to find. Most existing corpora either provide only summaries without full data sources (Hermann et al., 2015; Fabbri et al., 2019), or focus on narrow domains with limited document diversity (Cohan et al., 2018; Dasigi et al., 2021). Even benchmarks that include source documents, such as GovReport (Huang et al., 2021) or QMSum (Zhong et al., 2021), are limited to relatively clean, well-structured input formats rather than multimodal and often noisy collections found in real workflows. To address these gaps, we reverse-engineer task instructions $\mathcal{T}$ and source document sets $\mathcal{S}$ from web-based professional reports, which we treat as reference outputs $\mathcal{R}$.

➤ **Reference output document** $\mathcal{R}$: We collect 20 $\mathcal{R}$s (Table 1) via (1) LLM-assisted brainstorming with GPT-5 and (2) programmatic web search with SerpAPI to identify reports with reputable sources (e.g., GAO & Pew reports) versions spanning multiple years and categories (§A.3).

➤ **Source documents** $\mathcal{S}$: We assume that source references appear in each $\mathcal{R}$ as *explicit* and *implicit* references. *Explicit references* include citations,

Benchmark	Auto	MM	Eval	Human effort / cost (as reported)
BrowseComp-plus *(Chen et al., 2025)*	✓	✗	✗	Manual validation for 830 instances in ∼400 hours.
DeepResearchBench *(Du et al., 2025)*	✗	✗	✓	Manual task creation and evaluation of research reports (100 tasks).
Dolomites *(Malaviya et al., 2025)*	✓	✗	✗	Manual task creation (40 min / 2 tasks), validation (15 min / task), post-editing (20 min / example); $20 per task for 519 tasks.
ExpertLongBench *(Ruan et al., 2025)*	✗	✗	✓	Manual expert effort to collect tasks and write checklists for 1,050 samples.
GDPval *(Patwardhan et al., 2025)*	✗	✓	✗	Manual task construction (time and monetary cost not reported).
ResearchQA *(Yifei et al., 2025)*	✓	✗	✓	Automatic task generation: $350-2,185; checklist generation: $134-835 (varies by model).
AnalystBench	✓	✓	✓	Automatic task generation: $258.32; checklist generation: $62.80.

Table 2: Comparison of AnalystBench with related benchmarks. Auto: benchmark instances are auto-constructed; MM: supports multimodal generation; Eval: includes checklist-based evaluation.

hyperlinks, and direct mentions of documents or organizations. *Implicit references* correspond to uncited but source-specific content, such as distinctive statistics, quotations, figures, or characteristic phrasings. We first extract all explicit references from $\mathcal{R}$, including hyperlinks (via regular expressions) and named documents or entities (Figure 20), and retrieve the corresponding documents from the web using FireCrawl scraper.[4] For implicit references, we use GPT-5 to identify candidate phrases that mention external sources and issue web searches for each phrase via SerpAPI. The top retrieved results are automatically validated for relevance with GPT-4.1, with iterative query refinement applied when no suitable source is found. All validated links are then converted into machine-readable source documents, with web pages scraped and normalized into markdown and figures preserved inline as base64 strings.

➤ **Task instruction** $\mathcal{T}$: Given summaries of reference output and source documents, we prompt GPT-5 for a task description, which we manually review to be valid and consistent across $\mathcal{V}$.

2.3 Validating task feasibility

Automatic validation: We verify that $\mathcal{R}$ is grounded in its sources $\mathcal{S}$ using an automated claim verification pipeline (Ramu et al., 2024; Balasubramanian et al., 2025). Specifically, we decompose each $\mathcal{R}$ into verifiable claims (Song et al., 2024), and use GPT-5 to determine whether each claim is supported by $\mathcal{S}$ or other claims in the same $\mathcal{R}$.[5] We report a *coverage rate*, defined as the percentage of claims grounded in source documents. Overall, coverage across tasks exceeds 90% (Figure 5). Ungrounded claims often involve ordinal comparisons or reasoning over structured data (Table 6), which we manually verify when possible.[6] Claims that are unsupported even with supplemental documents are placed in an *excluded* section and discounted during evaluation.[7] Additionally, 31.25% of source documents are relevant but not necessary for producing the final report, which requires models to sift through non-essential materials (Figure 6).

Manual validation: We further validate task feasibility through a human study with 12 domain experts (Appendix C). Tasks span four domains (*Finance & Economics, Statistical & Data Analysis, Legislative & Justice, and Public & Government Policies*) with three experts each evaluating one variant per task. Given the task instruction $\mathcal{T}$ and source documents $\mathcal{S}$, experts judge whether a professional could complete the task using only these materials. Overall, 90% of tasks are judged fully feasible, and the remaining 10% are feasible with minor modifications (Table 8), typically involving instruction clarity or the effort required to process large document collections. We incorporate this feedback into the final benchmark.

3 Generating long-form documents

We consider multiple generation configurations, from lightweight workflows relying on a model's

[4] https://www.firecrawl.dev

[5] We mask the target sentence during verification to avoid self-matching and manually audit 50 random claims.

[6] We manually verified 137 of the 156 ungrounded claims (87%) by searching the source document for relevant evidence. The remaining 19 (13%) could not be verified from the released sources because they required reasoning over structured data (e.g., tables or computations) and programmatic access.

[7] Excluded section includes information not derivable from $\mathcal{S}$ (e.g., author details) or requiring human expertise (e.g., subjective recommendations).

parametric knowledge or simple chunking strategies, to more complex setups with code agents.

3.1 Generation methods

We evaluate one sample per model under two different approaches (see subsection D.2 for the stability justification).

Vanilla generation: We study the setting in which users simply "dump" input documents into the model's context window. When source documents exceed the available window size, we break them into manageable chunks, which are processed and then aggregated by the LLMs. All documents are converted into markdown and included in the context.[8] Statistics in large input tables (e.g., `.xlsx`, `.csv`) are summarized, and a preview of the table rows is included.

Coding agents: We evaluate OpenHands, a model-agnostic open-source coding agent framework that performs strongly on benchmarks such as TerminalBench (Team, 2025). Coding agents have the advantage of programmatically exploring full datasets, rather than depending on summarized inputs as in vanilla generation.

3.2 Ablation experiments

Varying model size: To study how model size affects generation quality, we evaluate two smaller Qwen variants: Qwen3-8B and Qwen3-14B.

Varying input components: In addition to the standard input configuration with task description, source documents, and table of contents, we evaluate three additional input configurations:

➤ **Task description $\mathcal{T}$:** This setting allows us to understand if the LLMs could use their parametric knowledge to generate a report without additional input sources or formatting guidance.

➤ **Task description $\mathcal{T}$ + Source documents $\mathcal{S}$:** This configuration is also realistic, as not all users will provide the LLMs with formatting guidelines like in the standard setting with a table of contents.

➤ **Task description $\mathcal{T}$ + Source documents $\mathcal{S}$ + Template document $\mathcal{R}^*$:** In this case, the model also receives a template document, which is the reference output document of another task variant.

[8]This reflects a real-world constraint where there is a limit on the number of files being uploaded to the API.

3.3 Default setup

We evaluate four frontier models: two *closed-source* (GPT-5.1, Gemini-3-Pro) and two *open-source* (Qwen3-32B, DeepSeek-R1).[9] Models are given structural guidance via a table of contents extracted from the reference output, specifying required sections, TL;DRs, and visualizations.[10] We use OpenHands as the default configuration, as it enables querying of full structured data rather than summarized context, unlike vanilla generation. Web search is disabled because all relevant sources are provided (§2.3), and enabling search could allow models to trivially retrieve the reference output, which compromises evaluation integrity.

4 Evaluating generated documents

We evaluate generated reports on overall quality (§4.1), groundedness in the source documents (§4.2), and generated figure quality (§4.3).

4.1 Checklist-based evaluation

Following Wadhwa et al. (2025) and Lee et al. (2025), we use task-specific checklists to evaluate long-form reports, where traditional automatic metrics are unreliable and full human evaluation is hard to scale (Xu et al., 2023). Each checklist consists of objective criteria derived from the task instruction $\mathcal{T}$ and reference output $\mathcal{R}$.

Checklist construction: For each task, we prompt GPT-5.1 with $\mathcal{T}$, $\mathcal{R}$, and excluded sections of $\mathcal{R}$ to generate 15 objective, task-grounded criteria that capture both high-level and fine-grained aspects of report quality.[11] Each checklist includes criteria assessing (i) instruction satisfaction, (ii) global coherence and tone, (iii) structural completeness, and (iv) factual details and statistics (Figure 15). The checklist is iteratively refined until the reference document satisfies all criteria to ensure alignment with the task and groundedness in $\mathcal{R}$.

Checklist validation: We validate the LLM-generated checklists against expert-authored rubrics for the same tasks (§C). Experts rate most checklists as clear and comprehensive or at least partially so (93%), with strong task suitability

[9]See Table 7 for model versions and hyperparameters.

[10]We manually audit 50 tables of contents to ensure appropriate detail and prevent information leakage.

[11]We generate three checklists with 50 criteria each for three tasks and find that 15 criteria provide sufficient coverage. Prior work uses about 7 criteria (Yifei et al., 2025) and 17 on average (Ruan et al., 2025).

(86%) and alignment with expert rubrics (84%) (Figure 13). Experts comment that the checklists focus on factual coverage and specificity, but less so on higher-level discourse qualities (e.g., organization or readability), and do not rank criterion importance. Therefore, the checklists are best suited for measuring coverage and factuality rather than subjective writing quality.

Scoring: Each generated report is evaluated by GPT-5.1, which assigns a 1-5 score for every criterion using explicit scoring guidelines to ensure consistency across evaluations (Figure 16).[12]

4.2 Groundedness evaluation

The checklist evaluation checks whether the document details are accurate and appropriate, but they do not directly measure how often models make up information not present in $\mathcal{S}$. We therefore follow the procedure in §2.3 and compute a hallucination rate, which represents the proportion of claims in the generated report that are not supported by $\mathcal{S}$.

4.3 Generated figure evaluation

We evaluate figures at a finer granularity using LLM judges. Generated figures (extracted from markdown paths, executable code, or base64 strings) are matched to reference figures in $\mathcal{R}$ based on caption similarity between each figure pair.[13] Each pair is scored by GPT-5.1 on readability, groundedness, utility, and reproducibility (Figure 21). We report visual quality score (normalized aggregate across dimensions) and coverage score, defined as the percentage of figures in the reference document matched by at least one generated figure.

5 Experiments and Results

Our results show that agent-based generation improves end-to-end report quality only for strong closed-source models; for open-source models, it often hurts performance, while simpler vanilla generation remains competitive (§5.1). Richer inputs, such as structural guidance via template documents or table of contents, improve quality and visualization (§5.2). Experts find the outputs grounded and readable but not operationally ready, citing missing key takeaways, underspecified quantitative facts, weak visuals, and limited actionability (§5.4).

5.1 Agent-based generation mainly benefits strong closed-source models

System-level performance: Agent-based generation outperforms vanilla generation only for strong closed-source models (Table 3). GPT-5.1 and Gemini-3-Pro both benefit from the OpenHands implementation, with GPT-5.1 achieving the highest checklist score overall (79.51%). In contrast, OpenHands hurts the checklist scores of open-source models: DeepSeek-R1 and Qwen3-32B drop by 2.57 and 4.56 percentage points relative to vanilla generation. Notably, under vanilla generation, open-source outperforms closed-source models (DeepSeek-R1: 60.24% vs. GPT-5.1: 58.24%). This pattern extends to visual metrics. Agent-based GPT-5.1 improves visual quality (+11.81) and coverage (+39), whereas open-source models cannot produce figures: coverage is $<$15% for DeepSeek-R1 and near zero for Qwen3-32B agents.

Task-level performance: We observe a relationship between input complexity and overall performance (Figure 2). Executive summary tasks, those with a structured format that require abstractive summarization rather than deeper analysis, achieve the highest checklist scores of over 90% (e.g., `acts`, `bjp`, `fed`). Performance degrades slightly on data-driven analytical tasks that require aggregation and statistical interpretation, with scores from 76.8 to 88% (e.g., `fred`, `oecd`, `pew`). In contrast, tasks that require long-horizon synthesis over large input collections perform substantially worse: `regents` and `columbus`, which involve millions of input tokens, score 25.3% and 41.1%, respectively.

Why closed-source models benefit from agents: The differences between vanilla and agentic results could be explained by reporting *behavior*, rather than by writing quality. In the vanilla setting, where each model has direct access to the source chunks in the prompt, open-source models tend to fill every required section even when there is little evidence, whereas closed-source models will sometimes state that the sources do not contain the requested information. This caution penalizes closed-source models on checklist-based metrics. In the OpenHands setting, by contrast, the model must complete a multi-step tool workflow and successfully write the required output file. Closed-source models almost

[12] We rescore 100 OpenHands ToC tasks with Claude-4.5-Sonnet as an independent judge, using the same fixed checklists. Claude and GPT scores were highly consistent (Pearson r=0.812), with nearly identical mean final scores (79.04 for Claude vs. 79.51 for GPT-5.1), which suggests the results are not specific to GPT-5.1 as the evaluator.

[13] Semantic similarity is computed with SentenceTransformers `all-MiniLM-L6-v2` (Reimers and Gurevych, 2019).

Model	Method	Inputs	Evaluation metrics				
			Length tokens	Checklist score % (↑)	Visual quality (↑)	Visual coverage % (↑)	Hallucination % (↓)
Default setup: vanilla generation vs. coding agents							
GPT-5.1	Vanilla	$\mathcal{T}+\mathcal{S}+$ ToC	4061	58.24	70.93	27.53	7.74
	Agents	$\mathcal{T}+\mathcal{S}+$ ToC	6780	**79.51**	82.74	<u>66.53</u>	14.81
Gemini-3-Pro	Vanilla	$\mathcal{T}+\mathcal{S}+$ ToC	1407	56.51	79.64	25.40	20.16
	Agents	$\mathcal{T}+\mathcal{S}+$ ToC	3307	71.04	83.25	62.87	16.84
DeepSeek-R1	Vanilla	$\mathcal{T}+\mathcal{S}+$ ToC	1262	60.24	73.68	15.87	**7.15**
	Agents	$\mathcal{T}+\mathcal{S}+$ ToC	1049	57.67	80.85	13.81	10.28
Qwen3-32B	Vanilla	$\mathcal{T}+\mathcal{S}+$ ToC	1455	59.33	83.47	4.76	11.88
	Agents	$\mathcal{T}+\mathcal{S}+$ ToC	1029	54.77	80.00	0.07	7.56
Ablation: scaling model size							
Qwen3-8B	Agents	$\mathcal{T}+\mathcal{S}+$ ToC	1468	54.95	N/A	0.00	17.08
Qwen3-14B	Agents	$\mathcal{T}+\mathcal{S}+$ ToC	1289	55.68	N/A	0.00	<u>7.43</u>
Ablation: varying input components							
GPT-5.1	–	$\mathcal{T}$	1337	44.27	N/A	0.00	14.58
GPT-5.1	Agents	$\mathcal{T}+\mathcal{S}$	6106	73.11	**84.81**	**72.95**	10.44
GPT-5.1	Agents	$\mathcal{T}+\mathcal{S}+\mathcal{R}^*$	6130	<u>75.09</u>	<u>84.39</u>	60.11	9.68

Table 3: Results for generation configurations and ablations. Metrics include checklist score, hallucination rate, and visual quality/coverage. Visual quality is the average of LLM judgments (1-5) aggregated over utility, groundedness, readability, and reproducibility; visual coverage is the % of figures in the reference document matched by at least one generated figure. **Bolded** and <u>underlined</u> denote best and second-best per metric.

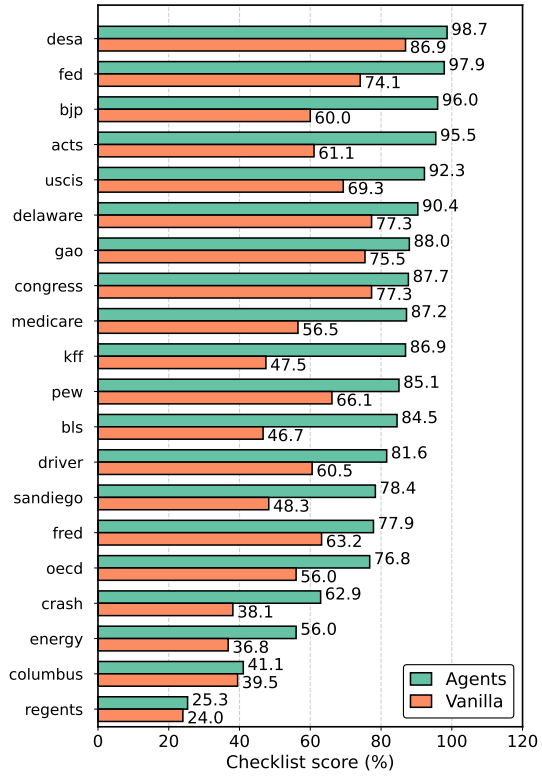

Figure 2: Checklist scores by task for GPT-5.1. Executive summary tasks score higher than long-horizon tasks with larger source collections.

always finish this end-to-end loop, while open-source models more frequently produce incomplete outputs or fail silently (e.g., a report with the required structure but placeholder content), which results in substantially shorter and less comprehensive reports. On the `fed/2024` variant, for instance, OpenHands GPT-5.1 produces 5,053 words while Qwen3-8B produces only 459, which is reflected in their respective checklist scores of 93.33% vs. 57.33%. Therefore, vanilla performance primarily reflects writing behavior (assertive vs. cautious coverage of required sections), whereas OpenHands performance primarily reflects whether a model can reliably complete the full tool loop end-to-end.

5.2 Scaling models reduces hallucination; additional inputs improves overall quality

Scaling model size: In the agent-based setting, moving from Qwen3-8B to 14B marginally increases the checklist score (+0.73) while substantially reducing hallucination (-9.65), yet neither model produces usable figures, as indicated by the N/A scores. Moving from 14B to 32B again results in little change in checklist performance. Looking at the execution logs and generated reports, we observe that Qwen3-32B often fails silently: it produces a report with the right headings and overall structure, but much of the actual content is missing or left as placeholders. The smaller 8B and 14B variants, by contrast, more often fail with ex-

plicit error messages, which allows the pipeline to be rerun until a report is produced (see Table 10 for examples of explicit execution errors). A likely root cause for this failure is context-window handling. OpenHands uses a fixed 118K-token context window for Qwen, whereas several tasks have input sources that substantially exceed this budget (see Table 1 for input sizes). The agentic loop continuously accumulates tool-call history and intermediate reasoning, and larger models such as 32B tend to produce more verbose planning trace, which exhaust the context budget earlier and can trigger premature termination, at which point the model may only emit the "skeleton" outline that later turns were meant to populate. This hypothesis is consistent with the stronger vanilla performance of the same model: the vanilla pipeline chunks sources to fit within the context window and is therefore more robust to large inputs. Accordingly, Qwen3-32B performs substantially better in the vanilla setting than in OpenHands (e.g., on `delawarema/autumn23`, 45.3% for 32B-OpenHands vs. 86.7% for 32B-vanilla).

Additional inputs: Moving from task-only input ($\mathcal{T}$) to also providing source documents ($\mathcal{T}$+$\mathcal{S}$) increases the checklist score by +28.84, improves figure generation (84.81% visual quality; 72.95% coverage), and reduces hallucination by 4.14 points. Notably, even without any source documents, the task-only setting still scores 44.27% on checklists, suggesting that GPT-5.1 has parametric knowledge of the report structures these tasks expect. The hallucination rate in this setting is also lower than expected; on closer inspection, these reports contain relatively few verifiable claims, so fewer statements are checked for grounding, which depresses the hallucination rate. Finally, adding the template document $\mathcal{R}^*$ marginally improves text quality and grounding but reduces visual coverage (-12.84), suggesting that a structured template like tables of contents may be more useful than an unstructured one that requires the model to infer what to include.

5.3 Agents follow a scan-and-write workflow

Agent execution logs show a common tool-use trajectory: `glob` (list files) → `terminal` (inspect/run commands) → `grep` (keyword scan) → `planning editor` (draft plan/checklist) → `file editor` (make edits) → `task tracker` (log status/next steps). The early stages look similar to a human's report-writing workflow, as analysts also start by listing sources, skimming, and using keyword searches to scope the report content. Nevertheless, the agent workflow relies more on skimming (`glob+grep`) than processing documents in depth. After the initial pass, they typically move straight into planning and editing with little evidence of returning to sources during drafting. In contrast, human workflows are more iterative: revisiting documents to extract and verify evidence, refine the outline, and revise.

5.4 Gaps in operational readiness

To evaluate more subjective qualities such as coherence and readability, we ask four domain experts (§C) to review GPT-5.1 reports generated with our default setting and provide free-form comments. We pair 20 instances of expert feedback with our checklist results[14] and code them into a qualitative taxonomy to identify recurring issues in generated reports (Table 4). Overall, experts describe the reports as careful and clear, but not yet ready for real-world use. The reports are generally grounded in the provided sources and explicitly note when evidence is missing. However, there are several gaps between these outputs and human-written professional reports. First, reports often fail to present the key takeaways up front and instead scatter throughout the text, making them difficult to quickly scan. Second, reports frequently omit or underspecify quantitative facts. For example, in the `medicare` task, experts note that summary statistics like depletion dates and payable percentages are absent, which reduces the value of the output as a fact sheet. Third, even when the content is reasonable, presentation choices reduce usability: for example, in `kff`, percentages are sometimes reported without clear baselines or references, making trends hard to interpret. Finally, experts point to weak actionability: many reports read more like background briefings than decision-support tools, and they do not clearly explain how the findings should inform real-world choices or next steps.

6 Related work

Our benchmark builds on work evaluating LLM agents on realistic end-to-end tasks and on domain-specific long-form generation grounded in external sources (e.g., databases or the open web; Table 2).

[14] We focus on criteria that score below 3/5 for each model.

Failure mode	Description	Short examples	Models
Inappropriate structure	Missing required sections or suboptimal section order.	Missing heading (e.g., "Overview"); required headings replaced with custom ones; extra promotional sections (e.g., firm resources, social media).	
Missing/incorrect tables	Required tables are absent, not parsable, or violate schema.	Required table in GAO reports replaced by plain text or bullet points.	
Missing/incorrect figures	Required charts are absent or fail spec (axes, labels, chart type).	Missing time-series or bar charts for criminal victimization in BJP tasks.	
Missing quant. facts	Omits or misstates dates, numeric/monetary values, or units.	Numeric placeholders instead of actual values (e.g., "XX%"); rounding errors (e.g., 45.5% rounded to 50% instead of 46%).	
Missing/incorrect citations	Omits or misassigns required identifiers.	Legal act citations without specific number or section (e.g., "PA" instead of "PA-121").	
Undefined terminology	Unclear acronyms or technical terms without prior definitions.	Acronyms such as CHEAPR/GWSA/NRES/SCEF used without expansion.	
Missing analysis	Omits required comparisons or historical/trend interpretation.	Reports values without comparison to previous years; omits legal grounding.	
Low actionability	Fails to translate findings into implications, affected parties, and next steps.	Missing "implications" subsection; listed changes but no concrete implementation steps.	
Scope and nuance errors	Blurs scope or strength of evidence; overgeneralized takeaways.	Oversimplifies complex trends; omits subgroup differences.	

Table 4: Taxonomy of common failure modes, with descriptions, representative examples, and the models on which each mode was observed (detected by human experts and checklist evaluation).

6.1 Realistic LLM agent benchmarks

LLM agents have been benchmarked within realistic professional environments. These settings include web, desktop, and mobile apps (Zhou et al., 2024; Deng et al., 2023; de Chezelles et al., 2025; Xie et al., 2024; Trivedi et al., 2024); multi-environment suites (Liu et al., 2025; Mialon et al., 2023); and domain-specific work software such as CRM (Huang et al., 2025a,b), ServiceNow (Drouin et al., 2024; Boisvert et al., 2025), and generic office workflows (Styles et al., 2024; Wang et al., 2024). A similar line of work target economically valuable tasks or those that are ripe for automation (Patwardhan et al., 2025; Mazeika et al., 2025; Jimenez et al., 2024; Yao et al., 2024; Barres et al., 2025; Miyai et al., 2025). Unlike prior work that builds tasks in controlled sandbox settings or requires expert effort, our benchmark instead mines recurring report tasks from the web, which makes it cheaper to scale and continually extensible.

6.2 Domain-specific long-form evaluation

Recent work has evaluated long-form generation in settings where models must process large external information sources, including database-style "deep research" environments (Coelho et al., 2025; FutureSearch et al., 2025; Sharma et al., 2025) and the web (Wei et al., 2025; Chen et al., 2025; Gou et al., 2025). Another line of work focuses on long-form generation itself, including Wikipedia-style articles (Shao et al., 2024), multi-domain long-form texts (Pham et al., 2024; Malaviya et al., 2025; Ruan et al., 2025), and scholarly articles grounded in academic sources (Yifei et al., 2025; Li et al., 2025a). Because these outputs are lengthy and only partially specified, evaluations typically rely on checklists. Therefore, we use an expert-validated checklists and groundedness metric to understand the quality and factuality of generated reports.

7 Conclusion

We introduce `AnalystBench`, a benchmark for end-to-end report generation grounded in large multimodal collections. `AnalystBench` enables systematic and customizable evaluation of report-writing workflows. We find that LLMs perform well on executive summaries but degrade on tasks that require long-horizon synthesis. Agent-based generation helps only strong closed-source models, while open-source models remain competitive with vanilla generation. Expert review identifies failure modes in structure, completeness, presentation, and actionability. Overall, `AnalystBench` highlights both progress and remaining limitations in report writing with LLMs.

Limitations

Task representativeness: AnalystBench includes 20 tasks across professional domains, but it does not capture every real-world analytical workflow. Since the benchmark focuses on publicly available reports that are mined from the web, there might be biases in terms of task selection toward government, policy, and economic tasks. As a result, many industry workflows, particularly those that are proprietary, highly collaborative, or require an internal organizational tool, are not represented.

Criteria weighting: Our checklist evaluation treats all criteria equally and does not have weighting to reflect differing levels of importance, which diverges from real-world settings where certain requirements are more important than others.

Lack of reproducible environment: Unlike benchmarks that operate in controlled sandboxes, our tasks use real-world documents that may change or become unavailable over time. While we provide tools for getting new variants, we cannot guarantee the long-term stability of all source materials, which might have to be obtained via Wayback Machine. We also disable web search during evaluation to prevent models from finding and relying on reference outputs, but this restriction may not reflect use cases where analysts do research on the Internet in addition to source documents.

Single-run evaluation: Due to the large amount of time required to run end-to-end report generation (OpenHands and MapReduce) over extremely long, multimodal inputs, we do not repeat the generation runs or try other decoding parameters. This lack of repeated generation may not account for result variability, especially for agentic implementation, where tool-use trajectories can differ across runs.

Ethical considerations

Copyright & usage statement: In line with our data release policy, we release only source URLs and reconstruction scripts rather than scraped document content. Researchers can use our evergreen variant tool to reconstruct source inputs locally from these URLs. Tasks drawing exclusively from U.S. government domains are public domain under 17 U.S.C. § 105 and will be made available immediately. For all remaining tasks, full URLs and reconstruction scripts will be released following legal review. Users are responsible for complying with the terms of service of each source domain when accessing materials. A contact for takedown requests is available at vmanjuna@adobe.com.

Use of LLM-generated reports: AnalystBench evaluates systems on generating professional reports, but benchmark outputs should be treated as drafts rather than ready-to-publish reports. We recommend that LLM-generated documents undergo human review for factual accuracy and appropriate interpretation, particularly in high-risk domains like finance and policy.

Risk of over-reliance: High-quality formatting and fluent prose can create an illusion of correctness and increase the risk that readers over-trust model outputs. Our evaluation shows there are still multiple failure modes in the generated reports (e.g., missing caveats, incorrect numbers, or misstatements). Users should therefore review the documents based on domain standards, including source checking, numerical verification, and accountability for final sign-off.

Sensitive data and private information: We construct tasks from publicly available documents and avoid including private information where we can. Nevertheless, public documents can still contain such sensitive information. Therefore, benchmark users should avoid unnecessary redistribution of sensitive details where possible.

AI usage disclosure: LLMs are used for writing assistance, not for generating the paper from scratch.

Acknowledgements

We thank Jen Le for her helpful feedback on benchmark design and HumanSignal for their support with the LabelStudio human annotation interface.

References

David H. Autor. 2015. Why are there still so many jobs? the history and future of workplace automation. *Journal of Economic Perspectives*, 29(3):3–30.

Sriram Balasubramanian, Samyadeep Basu, Koustava Goswami, Ryan Rossi, Varun Manjunatha, Roshan Santhosh, Ruiyi Zhang, Soheil Feizi, and Nedim Lipka. 2025. Decomposition-enhanced training for post-hoc attributions in language models. *Preprint*, arXiv:2510.25766.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. 2025. τ^2-bench: Evaluating conversational agents in a dual-control environment. *Preprint*, arXiv:2506.07982.

Léo Boisvert, Megh Thakkar, Maxime Gasse, Massimo Caccia, Thibault Le Sellier De Chezelles, Quentin Cappart, Nicolas Chapados, Alexandre Lacoste, and Alexandre Drouin. 2025. Workarena++: Towards compositional planning and reasoning-based common knowledge work tasks. *Preprint*, arXiv:2407.05291.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Erik Brynjolfsson, Danielle Li, and Lindsey R Raymond. 2023. Generative ai at work. Working Paper 31161, National Bureau of Economic Research.

Zijian Chen, Xueguang Ma, Shengyao Zhuang, Ping Nie, Kai Zou, Andrew Liu, Joshua Green, Kshama Patel, Ruoxi Meng, Mingyi Su, Sahel Sharifymoghaddam, Yanxi Li, Haoran Hong, Xinyu Shi, Xuye Liu, Nandan Thakur, Crystina Zhang, Luyu Gao, Wenhu Chen, and Jimmy Lin. 2025. Browsecomp-plus: A more fair and transparent evaluation benchmark of deep-research agent. *Preprint*, arXiv:2508.06600.

João Coelho, Jingjie Ning, Jingyuan He, Kangrui Mao, Abhijay Paladugu, Pranav Setlur, Jiahe Jin, Jamie Callan, João Magalhães, Bruno Martins, and Chenyan Xiong. 2025. Deepresearchgym: A free, transparent, and reproducible evaluation sandbox for deep research. *Preprint*, arXiv:2505.19253.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. *Preprint*, arXiv:1804.05685.

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

Thibault Le Sellier de Chezelles, Maxime Gasse, Alexandre Lacoste, Massimo Caccia, Alexandre Drouin, Léo Boisvert, Megh Thakkar, Tom Marty, Rim Assouel, Sahar Omidi Shayegan, Lawrence Keunho Jang, Xing Han Lù, Ori Yoran, Dehan Kong, Frank F. Xu, Siva Reddy, Graham Neubig, Quentin Cappart, Russ Salakhutdinov, and Nicolas Chapados. 2025. The browsergym ecosystem for web agent research. *Transactions on Machine Learning Research*. Expert Certification.

Fabrizio Dell'Acqua, Edward McFowland III, Ethan R. Mollick, Hila Lifshitz-Assaf, Katherine Kellogg, Saran Rajendran, Lisa Krayer, François Candelon, and Karim R. Lakhani. 2023. Navigating the jagged technological frontier: Field experimental evidence of the effects of ai on knowledge worker productivity and quality. Technical Report Working Paper No. 24-013, Harvard Business School Technology & Operations Mgt. Unit and The Wharton School.

Chunyuan Deng, Yilun Zhao, Xiangru Tang, Mark Gerstein, and Arman Cohan. 2024. Investigating data contamination in modern benchmarks for large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8706–8719, Mexico City, Mexico. Association for Computational Linguistics.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. 2023. Mind2web: Towards a generalist agent for the web. In *Advances in Neural Information Processing Systems*, volume 36, pages 28091–28114. Curran Associates, Inc.

Jesse Dodge, Maarten Sap, Ana Marasović, William Agnew, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. 2021. Documenting large webtext corpora: A case study on the colossal clean crawled corpus. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1286–1305, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Alexandre Drouin, Maxime Gasse, Massimo Caccia, Issam H. Laradji, Manuel Del Verme, Tom Marty, David Vazquez, Nicolas Chapados, and Alexandre Lacoste. 2024. Workarena: how capable are web agents at solving common knowledge work tasks? In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Mingxuan Du, Benfeng Xu, Chiwei Zhu, Xiaorui Wang, and Zhendong Mao. 2025. Deepresearch bench: A comprehensive benchmark for deep research agents. *Preprint*, arXiv:2506.11763.

Alexander Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir Radev. 2019. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1074–1084, Florence, Italy. Association for Computational Linguistics.

Alexander R. Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir

Radev. 2021. SummEval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

FutureSearch, Nikos I. Bosse, Jon Evans, Robert G. Gambee, Daniel Hnyk, Peter Mühlbacher, Lawrence Phillips, Dan Schwarz, and Jack Wildman. 2025. Deep research bench: Evaluating ai web research agents. *Preprint*, arXiv:2506.06287.

Boyu Gou, Zanming Huang, Yuting Ning, Yu Gu, Michael Lin, Weijian Qi, Andrei Kopanev, Botao Yu, Bernal Jiménez Gutiérrez, Yiheng Shu, Chan Hee Song, Jiaman Wu, Shijie Chen, Hanane Nour Moussa, Tianshu Zhang, Jian Xie, Yifei Li, Tianci Xue, Zeyi Liao, and 7 others. 2025. Mind2web 2: Evaluating agentic search with agent-as-a-judge. *Preprint*, arXiv:2506.21506.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Kung-Hsiang Huang, Akshara Prabhakar, Sidharth Dhawan, Yixin Mao, Huan Wang, Silvio Savarese, Caiming Xiong, Philippe Laban, and Chien-Sheng Wu. 2025a. CRMArena: Understanding the capacity of LLM agents to perform professional CRM tasks in realistic environments. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3830–3850, Albuquerque, New Mexico. Association for Computational Linguistics.

Kung-Hsiang Huang, Akshara Prabhakar, Onkar Thorat, Divyansh Agarwal, Prafulla Kumar Choubey, Yixin Mao, Silvio Savarese, Caiming Xiong, and Chien-Sheng Wu. 2025b. Crmarena-pro: Holistic assessment of llm agents across diverse business scenarios and interactions. *Preprint*, arXiv:2505.18878.

Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. 2021. Efficient attentions for long document summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1419–1436, Online. Association for Computational Linguistics.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. 2024. Swe-bench: Can language models resolve real-world github issues? In *International Conference on Learning Representations (ICLR)*.

Shankar Kantharaj, Rixie Tiffany Leong, Xiang Lin, Ahmed Masry, Megh Thakkar, Enamul Hoque, and Shafiq Joty. 2022. Chart-to-text: A large-scale benchmark for chart summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4005–4023, Dublin, Ireland. Association for Computational Linguistics.

Yukyung Lee, JoongHoon Kim, Jaehee Kim, Hyowon Cho, Jaewook Kang, Pilsung Kang, and Najoung Kim. 2025. CheckEval: A reliable LLM-as-a-judge framework for evaluating text generation using checklists. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 15782–15809, Suzhou, China. Association for Computational Linguistics.

Minghao Li, Ying Zeng, Zhihao Cheng, Cong Ma, and Kai Jia. 2025a. Reportbench: Evaluating deep research agents via academic survey tasks. *Preprint*, arXiv:2508.15804.

Sitong Li, Stefano Padilla, Pierre Le Bras, Junyu Dong, and Mike Chantler. 2025b. A review of llm-assisted ideation. *Preprint*, arXiv:2503.00946.

Shuaiqi Liu, Jiannong Cao, Ruosong Yang, and Zhiyuan Wen. 2022. Long text and multi-table summarization: Dataset and method. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1995–2010, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, and 3 others. 2025. Agentbench: Evaluating llms as agents. *Preprint*, arXiv:2308.03688.

Chaitanya Malaviya, Priyanka Agrawal, Kuzman Ganchev, Pranesh Srinivasan, Fantine Huot, Jonathan Berant, Mark Yatskar, Dipanjan Das, Mirella Lapata, and Chris Alberti. 2025. Dolomites: Domain-specific long-form methodical tasks. *Transactions of the Association for Computational Linguistics*, 13:1–29.

Mantas Mazeika, Alice Gatti, Cristina Menghini, Udari Madhushani Sehwag, Shivam Singhal, Yury Orlovskiy, Steven Basart, Manasi Sharma, Denis Peskoff, Elaine Lau, Jaehyuk Lim, Lachlan Carroll, Alice Blair, Vinaya Sivakumar, Sumana Basu, Brad Kenstler, Yuntao Ma, Julian Michael, Xiaoke Li, and 28 others. 2025. Remote labor index: Measuring ai automation of remote work. *Preprint*, arXiv:2510.26787.

Grégoire Mialon, Clémentine Fourrier, Craig Swift, Thomas Wolf, Yann LeCun, and Thomas Scialom. 2023. Gaia: a benchmark for general ai assistants. *Preprint*, arXiv:2311.12983.

Atsuyuki Miyai, Zaiying Zhao, Kazuki Egashira, Atsuki Sato, Tatsumi Sunada, Shota Onohara, Hiromasa Yamanishi, Mashiro Toyooka, Kunato Nishina, Ryoma Maeda, Kiyoharu Aizawa, and Toshihiko Yamasaki. 2025. Webchorearena: Evaluating web browsing agents on realistic tedious web tasks. *Preprint*, arXiv:2506.01952.

Ramesh Nallapati, Bowen Zhou, Cicero Nogueira dos santos, Caglar Gulcehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *Preprint*, arXiv:1602.06023.

Ankur P. Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. 2020. Totto: A controlled table-to-text generation dataset. *Preprint*, arXiv:2004.14373.

Tejal Patwardhan, Rachel Dias, Elizabeth Proehl, Grace Kim, Michele Wang, Olivia Watkins, Simón Posada Fishman, Marwan Aljubeh, Phoebe Thacker, Laurance Fauconnet, Natalie S. Kim, Patrick Chao, Samuel Miserendino, Gildas Chabot, David Li, Michael Sharman, Alexandra Barr, Amelia Glaese, and Jerry Tworek. 2025. Gdpval: Evaluating ai model performance on real-world economically valuable tasks. *Preprint*, arXiv:2510.04374.

Chau Minh Pham, Simeng Sun, and Mohit Iyyer. 2024. Suri: Multi-constraint instruction following in long-form text generation. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 1722–1753, Miami, Florida, USA. Association for Computational Linguistics.

Ratish Puduppully, Li Dong, and Mirella Lapata. 2019. Data-to-text generation with content selection and planning. *Preprint*, arXiv:1809.00582.

Pritika Ramu, Koustava Goswami, Apoorv Saxena, and Balaji Vasan Srinivasan. 2024. Enhancing post-hoc attributions in long document comprehension via coarse grained answer decomposition. *Preprint*, arXiv:2409.17073.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Jie Ruan, Inderjeet Nair, Shuyang Cao, Amy Liu, Sheza Munir, Micah Pollens-Dempsey, Tiffany Chiang, Lucy Kates, Nicholas David, Sihan Chen, Ruxin Yang, Yuqian Yang, Jasmine Gump, Tessa Bialek, Vivek Sankaran, Margo Schlanger, and Lu Wang. 2025. Expertlongbench: Benchmarking language models on expert-level long-form generation tasks with structured checklists. *Preprint*, arXiv:2506.01241.

Oscar Sainz, Jon Campos, Iker García-Ferrero, Julen Etxaniz, Oier Lopez de Lacalle, and Eneko Agirre. 2023. NLP evaluation in trouble: On the need to measure LLM data contamination for each benchmark. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10776–10787, Singapore. Association for Computational Linguistics.

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

Manasi Sharma, Chen Bo Calvin Zhang, Chaithanya Bandi, Clinton Wang, Ankit Aich, Huy Nghiem, Tahseen Rabbani, Ye Htet, Brian Jang, Sumana Basu, Aishwarya Balwani, Denis Peskoff, Marcos Ayestaran, Sean M. Hendryx, Brad Kenstler, and Bing Liu. 2025. Researchrubrics: A benchmark of prompts and rubrics for evaluating deep research agents. *Preprint*, arXiv:2511.07685.

Ben Shneiderman. 2020. Human-centered artificial intelligence: Reliable, safe & trustworthy. *Preprint*, arXiv:2002.04087.

Yixiao Song, Yekyung Kim, and Mohit Iyyer. 2024. VeriScore: Evaluating the factuality of verifiable claims in long-form text generation. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 9447–9474, Miami, Florida, USA. Association for Computational Linguistics.

Olly Styles, Sam Miller, Patricio Cerda-Mardini, Tanaya Guha, Victor Sanchez, and Bertie Vidgen. 2024. Workbench: a benchmark dataset for agents in a realistic workplace setting. *Preprint*, arXiv:2405.00823.

Benny J. Tang, Angie Boggust, and Arvind Satyanarayan. 2023. VisText: A Benchmark for Semantically Rich Chart Captioning. In *The Annual Meeting of the Association for Computational Linguistics (ACL)*.

The Terminal-Bench Team. 2025. Terminal-bench: A benchmark for ai agents in terminal environments.

Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, Edward Li, Shashank Gupta, Ashish Sabharwal, and Niranjan Balasubramanian. 2024. Appworld: A controllable world of apps and people for benchmarking interactive coding agents. *Preprint*, arXiv:2407.18901.

Manya Wadhwa, Zayne Rea Sprague, Chaitanya Malaviya, Philippe Laban, Junyi Jessy Li, and Greg Durrett. 2025. Evalagents: Discovering implicit evaluation criteria from the web. In *Second Conference on Language Modeling*.

Xingyao Wang, Boxuan Li, Yufan Song, Frank F. Xu, Xiangru Tang, Mingchen Zhuge, Jiayi Pan, Yueqi Song, Bowen Li, Jaskirat Singh, Hoang H. Tran, Fuqiang Li, Ren Ma, Mingzhang Zheng, Bill Qian, Yanjun Shao, Niklas Muennighoff, Yizhe Zhang, Binyuan Hui, and 5 others. 2025. Openhands: An open platform for AI software developers as generalist agents. In *The Thirteenth International Conference on Learning Representations*.

Zilong Wang, Yuedong Cui, Li Zhong, Zimin Zhang, Da Yin, Bill Yuchen Lin, and Jingbo Shang. 2024. Officebench: Benchmarking language agents across multiple applications for office automation. *Preprint*, arXiv:2407.19056.

Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia

Glaese. 2025. Browsecomp: A simple yet challenging benchmark for browsing agents. *Preprint*, arXiv:2504.12516.

Sam Wiseman, Stuart M. Shieber, and Alexander M. Rush. 2017. Challenges in data-to-document generation. *Preprint*, arXiv:1707.08052.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. 2024. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. *Preprint*, arXiv:2404.07972.

Fangyuan Xu, Yixiao Song, Mohit Iyyer, and Eunsol Choi. 2023. A critical evaluation of evaluations for long-form question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3225–3245, Toronto, Canada. Association for Computational Linguistics.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. 2024. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. *Preprint*, arXiv:2406.12045.

Li S. Yifei, Allen Chang, Chaitanya Malaviya, and Mark Yatskar. 2025. Researchqa: Evaluating scholarly question answering at scale across 75 fields with survey-mined questions and rubrics. *Preprint*, arXiv:2509.00496.

Ming Zhong, Da Yin, Tao Yu, Ahmad Zaidi, Mutethia Mutuma, Rahul Jha, Ahmed Hassan Awadallah, Asli Celikyilmaz, Yang Liu, Xipeng Qiu, and Dragomir Radev. 2021. Qmsum: A new benchmark for query-based multi-domain meeting summarization. *Preprint*, arXiv:2104.05938.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. 2024. Webarena: A realistic web environment for building autonomous agents. *Preprint*, arXiv:2307.13854.

A Constructing `AnalystBench`: additional details

A.1 List of data sources

Table 5 shows the sources included in `AnalystBench`.

Task (category / instance)	Output document (gold reference URL)	Input source domains (top-level)
Connecticut Acts Affecting Environment		acts_summary/
2021	https://www.cga.ct.gov/olr/Documents/year/AA/2021AA-0111_2021%20Acts%20Affecting%20Environment.pdf	ct.gov
2022	https://www.cga.ct.gov/olr/Documents/year/AA/2022AA-0090_2022%20Acts%20Affecting%20Environment.pdf	ct.gov
2023	https://www.cga.ct.gov/olr/Documents/year/AA/2023AA-0120_2023%20Acts%20Affecting%20Environment.pdf	ct.gov
2024	https://www.cga.ct.gov/olr/Documents/year/AA/2024AA-0083_2024%20Acts%20Affecting%20Environment.pdf	ct.gov
2025	https://www.cga.ct.gov/olr/Documents/year/AA/2025AA-0110_2025%20Acts%20Affecting%20Environment.pdf	ct.gov
BJS Criminal Victimization		bjp_summary/
2019	https://bjs.ojp.gov/content/pub/pdf/cv19_sum.pdf	ojp.gov
2020	https://bjs.ojp.gov/media/65336/download	ojp.gov
2021	https://bjs.ojp.gov/content/pub/pdf/cv21_sum.pdf	ojp.gov
2022	https://bjs.ojp.gov/document/cv22_sum.pdf	ojp.gov
2023	https://bjs.ojp.gov/document/cv23_sum.pdf	ojp.gov
BLS Employment Situation		bls_report/
0425	https://web.archive.org/web/20250601025039/https://www.bls.gov/news.release/empsit.nr0.htm	archive.org
0525	https://web.archive.org/web/20250629072946/https://www.bls.gov/news.release/empsit.nr0.htm	archive.org
0625	https://web.archive.org/web/20250709020338/https://www.bls.gov/news.release/empsit.nr0.htm	archive.org
0725	https://web.archive.org/web/20250805014822/https://www.bls.gov/news.release/empsit.nr0.htm	archive.org
0825	https://web.archive.org/web/20250930143450/https://www.bls.gov/news.release/empsit.nr0.htm	archive.org
Columbus OH Popular Annual Financial Report		columbus_report/
PAFR_2020	https:www.columbus.gov/files/sharedassets/city/v/1/city-auditor/pafr/2020_pafrfinal.pdf	Local news outlets and municipal data sources
PAFR_2021	https://www.columbus.gov/files/sharedassets/city/v/1/city-auditor/pafr/2021-pafr_final.pdf	Local news outlets and municipal data sources
PAFR_2022	https://www.columbus.gov/files/sharedassets/city/v/1/city-auditor/pafr/2022_pafr.pdf	Local news outlets and municipal data sources
PAFR_2023	https://www.columbus.gov/files/sharedassets/city/v/2/city-auditor/pafr/pafr-2023.pdf	Local news outlets and municipal data sources
PAFR_2024	https://www.columbus.gov/files/sharedassets/city/v/1/city-auditor/pafr/city-of-columbus-pafr-final-for-web-4-2-2025.pdf	Local news outlets and municipal data sources
CRS Congressional Court Watcher		congress_summary/
LSB11075.3	https://www.congress.gov/crs_external_products/LSB/PDF/LSB11075/LSB11075.3.pdf	epa.gov, govinfo.gov, house.gov, supremecourt.gov, uscourts.gov
LSB11099.2	https://www.congress.gov/crs_external_products/LSB/PDF/LSB11099/LSB11099.2.pdf	alaskarailroad.com, house.gov, supremecourt.gov, uscourts.gov
LSB11109.3	https://www.congress.gov/crs_external_products/LSB/PDF/LSB11109/LSB11109.3.pdf	congress.gov, house.gov, justia.com, supremecourt.gov, texasattorneygeneral.gov, uscourts.gov
LSB11113.1	https://www.congress.gov/crs_external_products/LSB/PDF/LSB11113/LSB11113.1.pdf	archive.org, congress.gov, house.gov, newyorkconvention.org, supremecourt.gov, uscourts.gov
LSB11144.2	https://www.congress.gov/crs_external_products/LSB/PDF/LSB11144/LSB11144.2.pdf	archive.org, congress.gov, govinfo.gov, house.gov, loc.gov, supremecourt.gov, uscourts.gov
NHTSA Pedestrian Crash Report		crash_report/
DOT_813079	https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/813079	nhtsa.gov
DOT_813310	https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/813310	nhtsa.gov
DOT_813458	https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/813458	nhtsa.gov
DOT_813590	https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/813590	nhtsa.gov
DOT_813727	https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/813727	nhtsa.gov

continued on next page

(Table 5 continued)

Task (category / instance)	Output document (gold reference URL)	Input source domains (top-level)
Paul Weiss Delaware M&A Quarterly		delawarema_summary/
autumn23	Paul Weiss Delaware M&A Quarterly (commercial legal publication)	Major U.S. law firm client alerts and legal analyses
spring23	Paul Weiss Delaware M&A Quarterly (commercial legal publication)	Major U.S. law firm client alerts and legal analyses
spring24	Paul Weiss Delaware M&A Quarterly (commercial legal publication)	Major U.S. law firm client alerts and legal analyses
summer23	Paul Weiss Delaware M&A Quarterly (commercial legal publication)	Major U.S. law firm client alerts and legal analyses
winter23	Paul Weiss Delaware M&A Quarterly (commercial legal publication)	Major U.S. law firm client alerts and legal analyses
UN DESA Policy Brief		desa_report/
PB157	https://desapublications.un.org/file/20539/download	apc.org, brookings.edu, doi.org, gdpr.eu, gob.es, icnl.org, london.gov.uk, oecd.org, serviciocivil.cl, un.org, undp.org
PB164	https://desapublications.un.org/file/20744/download	doi.org, gov.gh, inff.org, un.org, weforum.org
PB174	https://desapublications.un.org/file/21159/download	doi.org, eduskunta.fi, gouv.fr, ibm.com, itu.int, oecd-opsi.org, pcdn.co, un-futureslab.org, un.org, undp.org, unep.org, valtioneuvosto.fi
PB177	https://desapublications.un.org/file/21224/download	development-finance.org, gao.gov, idi.no, internationalbudget.org, pefa.org, un.org
PB181	https://desapublications.un.org/file/21376/download	canada.ca, elsyn.gr, europa.eu, gao.gov, gc.ca, idi.no, intosai.org, oag.go.ug, tfma.eu, un.org, valstybeskontrole.lt
NILC Driver's License Access Table		driverlicense_summary/
2021	https://www.nilc.org/wp-content/uploads/2021/12/Drivers-license-access-table-2021-12-09-updated.pdf	aclu-md.org, aclusocal.org, billtrack50.com, ca.gov, civilbeat.org, colorado.gov, coloradoimmigrant.org, ct.gov, delaware.gov, dhs.gov, hawaii.gov, honolulu.gov, ilcatholic.org, ilga.gov, illegalprivilege.com, informedimmigrant.com, ksltv.com, lawinfo.com, legiscan.com, maryland.gov, mi.gov, microjuris.com, migrantjustice.net, ncsl.org, newmexico.gov, nilc.org, nj.gov, njoag.gov, njpp.org, nmlegis.gov, nv.gov, nv.us, ny.gov, nyassembly.gov, nysenate.gov, oregonlegislature.gov, passage.law, procon.org, rainn.org, skyscrapercity.com, state.gov, trackbill.com, tsa.gov, urban.org, utah.gov, vermont.gov, virginia.gov, votesmart.org, wa.gov, windows.net
2022	https://www.nilc.org/wp-content/uploads/2022/10/drivers-license-access-table-2022-09-30.pdf	archive.org, ca.gov, colorado.gov, ct.gov, delaware.gov, hawaii.gov, ilga.gov, malegislature.gov, maryland.gov, nj.us, nmlegis.gov, nv.us, nyassembly.gov, oslpr.org, rilegislature.gov, utah.gov, virginia.gov, vt.us, wa.gov
2023	https://www.nilc.org/wp-content/uploads/2023/07/drivers-license-access-table-2023-07-01-.pdf	aclu-md.org, aclusocal.org, albanylaw.edu, archive.org, billtrack50.com, ca.gov, caimmigrant.org, colorado.gov, coloradoimmigrant.org, ct.gov, dc.gov, de.gov, delaware.gov, dhs.gov, hawaii.gov, icirr.org, ilcatholic.org, ilga.gov, informedimmigrant.com, legiscan.com, malegislature.gov, maryland.gov, mn.gov, ncsl.org, newmexico.gov, nilc.org, njleg.gov, nmlegis.gov, nmrestaurants.org, nv.us, ny.gov, nysenate.gov, oregonlegislature.gov, pew.org, ri.gov, rilegislature.gov, senatorholmes.com, siu.edu, state.gov, trackbill.com, utah.gov, vermont.gov, virginia.gov, votesmart.org, wa.gov, wikipedia.org, windows.net, youtube.com
2024	https://www.nilc.org/wp-content/uploads/2024/06/drivers-license-access-table-06-2024pdf-1.pdf	aclu-md.org, aclusocal.org, adsuarlaw.com, billtrack50.com, ca.gov, caimmigrant.org, colorado.gov, ct.gov, dc.gov, de.gov, delaware.gov, digitaldemocracy.org, freeway.com, hawaii.gov, honolulu.gov, icirr.org, ilcatholic.org, ilga.gov, ilrc.org, informedimmigrant.com, justia.com, latinojustice.org, legiscan.com, malegislature.gov, maryland.gov, mi.gov, mn.gov, ncsl.org, newmexico.gov, nilc.org, nj.gov, njpp.org, nmlegis.gov, nmrestaurants.org, nv.us, ny.gov, nysenate.gov, oregonlegislature.gov, pew.org, pr.gov, ri.gov, rilegislature.gov, statebillinfo.com, urban.org, utah.gov, vermont.gov, virginia.gov, votesmart.org, wa.gov, wikipedia.org, windows.net
2025	https://www.nilc.org/wp-content/uploads/2024/06/drivers-license-access-table-2025-08.pdf	ca.gov, co.us, colorado.gov, ct.gov, dccouncil.gov, delaware.gov, hawaii.gov, ilga.gov, malegislature.gov, maryland.gov, mn.gov, nj.us, nmlegis.gov, nv.us, nyassembly.gov, oregonlegislature.gov, rilegislature.gov, utah.gov, vermont.gov, virginia.gov, vt.us, wa.gov
EIA Today in Energy		energy_report/
battery	https://www.eia.gov/todayinenergy/detail.php?id=66164#	eia.gov
co2	https://www.eia.gov/todayinenergy/detail.php?id=66104	eia.gov

continued on next page

(Table 5 continued)

Task (category / instance)	Output document (gold reference URL)	Input source domains (top-level)
electricity	https://www.eia.gov/todayinenergy/detail.php?id=66204#	eia.gov
heating	https://www.eia.gov/todayinenergy/detail.php?id=66324#	census.gov, eia.gov, energy.gov, klgates.com, spglobal.com
texas	https://www.eia.gov/todayinenergy/detail.php?id=66224#	eia.gov
Federal Reserve Economic Well-Being		fed_summary/
2020	https://www.federalreserve.gov/publications/2021-economic-well-being-of-us-households-in-2020-executive-summary.htm	federalreserve.gov
2021	https://www.federalreserve.gov/newsevents/pressreleases/files/other20220523a1.pdf	federalreserve.gov
2022	https://www.federalreserve.gov/newsevents/pressreleases/files/other20230522a1.pdf	federalreserve.gov
2023	https://www.federalreserve.gov/newsevents/pressreleases/files/other20240521a1.pdf	federalreserve.gov
2024	https://www.federalreserve.gov/newsevents/pressreleases/files/other20250528a1.pdf	federalreserve.gov
FRED Blog		fred_report/
credit	https://fredblog.stlouisfed.org/2025/09/the-ups-and-downs-in-credit-card-borrowing-lines/	bostonfed.org, federalreserve.gov, stlouisfed.org
european	https://fredblog.stlouisfed.org/2025/09/a-look-across-european-postal-prices/	happywhatever.nl, postnord.dk, stlouisfed.org, tcmb.gov.tr, tv5.com.tr, wikipedia.org
labor	https://fredblog.stlouisfed.org/2025/09/trends-in-us-labor-force-participation-rates-for-men/	aeaweb.org, brookings.edu, frbsf.org, repec.org
trade_balance	https://fredblog.stlouisfed.org/2025/09/the-trade-balance-the-dollar-and-trade-policy/	*publisher's own data files (CSV/XLSX)*
trends	https://fredblog.stlouisfed.org/2025/09/trends-in-the-us-distribution-of-net-worth/	repec.org
GAO Recommendations Letter		gao_summary/
gao-25-108460	https://www.gao.gov/assets/gao-25-108460.pdf	gao.gov
gao-25-108464	https://www.gao.gov/assets/gao-25-108464.pdf	gao.gov
gao-25-108478	https://www.gao.gov/assets/gao-25-108478.pdf	gao.gov
gao-25-108537	https://www.gao.gov/assets/gao-25-108537.pdf	cio.gov, federalregister.gov, gao.gov, govinfo.gov, whitehouse.gov
gao-25-108540	https://www.gao.gov/assets/gao-25-108540.pdf	cio.gov, gao.gov, splunk.com, whitehouse.gov
KFF Health Tracking Poll		kff_report/
authorization	https://www.kff.org/patient-consumer-protections/kff-health-tracking-poll-public-finds-prior-authorization-process-difficult-to-manage/	Major U.S. news outlets and health policy sources
bbb	https://www.kff.org/medicaid/kff-health-tracking-poll-views-of-the-one-big-beautiful-bill/	Major U.S. news outlets and health policy sources
glp24	https://www.kff.org/health-costs/kff-health-tracking-poll-may-2024-the-publics-use-and-views-of-glp-1-drugs/	Major U.S. news outlets and health policy sources
medicaid	https://www.kff.org/medicaid/kff-health-tracking-poll-the-publics-views-of-funding-reductions-to-medicaid/	Major U.S. news outlets and health policy sources
usaid	https://www.kff.org/global-health-policy/kff-health-tracking-poll-february-2025-the-publics-views-on-global-health-and-usaid/	Major U.S. news outlets and health policy sources
Medicare Trustees Fact Sheet		medicare_summary/
2020	https://home.treasury.gov/system/files/136/Fact-Sheet-TR20.pdf	cms.gov, ssa.gov
2021	https://home.treasury.gov/system/files/136/Fact-Sheet-2021-Social-Security-and-Medicare-Trustees-Reports.pdf	cms.gov
2022	https://home.treasury.gov/system/files/136/TR-2022-Fact-Sheet.pdf	cms.gov
2023	https://home.treasury.gov/system/files/136/TR-2023-Fact-Sheet.pdf	cms.gov
2024	https://home.treasury.gov/system/files/136/TR-2024-Fact-Sheet.pdf	cms.gov
OECD Revenue Statistics		oecd_report/
canada	https://www.oecd.org/content/dam/oecd/en/topics/policy-sub-issues/global-tax-revenues/revenue-statistics-canada.pdf	oecd.org

continued on next page

(Table 5 continued)

Task (category / instance)	Output document (gold reference URL)	Input source domains (top-level)
france	https://www.oecd.org/content/dam/oecd/en/topics/policy-sub-issues/global-tax-revenues/revenue-statistics-france.pdf	oecd.org
germany	https://www.oecd.org/content/dam/oecd/en/topics/policy-sub-issues/global-tax-revenues/revenue-statistics-germany.pdf	oecd.org
japan	https://www.oecd.org/content/dam/oecd/en/topics/policy-sub-issues/global-tax-revenues/revenue-statistics-japan.pdf	oecd.org
mexico	https://www.oecd.org/content/dam/oecd/en/topics/policy-sub-issues/global-tax-revenues/revenue-statistics-mexico.pdf	oecd.org
Pew Research Fact Sheet		pew_report/
news_platform	https://www.pewresearch.org/journalism/fact-sheet/news-platform-fact-sheet/	*publisher's own data files (CSV/XLSX)*
npors	https://www.pewresearch.org/methods/fact-sheet/national-public-opinion-reference-survey-npors/	pewresearch.org
party	https://www.pewresearch.org/politics/fact-sheet/party-affiliation-fact-sheet-npors/	pewresearch.org
podcasts	https://www.pewresearch.org/journalism/fact-sheet/podcasts-and-news-fact-sheet/	pewresearch.org
social_media	https://www.pewresearch.org/journalism/fact-sheet/social-media-and-news-fact-sheet/	pewresearch.org
NY Regents Audits/Budget Committee		regents_summary/
2013	https://www.regents.nysed.gov/sites/regents/files/1013audbfd1%5B2%5D.pdf	ny.gov, nysed.gov
2017	https://www.regents.nysed.gov/sites/regents/files/317audbfd1.pdf	ny.gov, ny.us, nyc.gov, nysed.gov, ocmboces.org
2018	https://www.regents.nysed.gov/sites/regents/files/618audbfd1.pdf	ny.gov, nysed.gov
2019a	https://www.regents.nysed.gov/sites/regents/files/419audbfd1.pdf	dasny.org, ichabodcrane.org, ny.gov, nysed.gov
2019b	https://www.regents.nysed.gov/sites/regents/files/119audbfd1.pdf	ed.gov, ny.gov, nyc.gov, nysed.gov
San Diego Capital Improvement Program		sandiego_report/
2021	https://www.sandiego.gov/sites/default/files/fy21ab_v1cip.pdf	sandag.org, sandiego.gov, voiceofsandiego.org
2022	https://www.sandiego.gov/sites/default/files/fy22ab_v1cip.pdf	sandag.org, sandiego.gov, voiceofsandiego.org, wsimg.com
2023	https://www.sandiego.gov/sites/default/files/fy23ab_v1cip.pdf	facebook.com, sandiego.gov
2024	https://www.sandiego.gov/sites/default/files/fy24ab_v1cip.pdf	insidesandiego.org, sandiego.gov, windows.net
2025	https://www.sandiego.gov/sites/default/files/2024-08/fy25ab_v1cip.pdf	sandiego.gov
USCIS Policy Alert		uscis_summary/
20250821-SubmissionOfFees	https://www.uscis.gov/sites/default/files/document/policy-manual-updates/20250821-SubmissionOfFees.pdf	ecfr.gov, govinfo.gov, uscis.gov
20250829-ElectronicPayments	https://www.uscis.gov/sites/default/files/document/policy-manual-updates/20250829-ElectronicPayments.pdf	ecfr.gov, govinfo.gov, house.gov, pay.gov, uscis.gov, usembassy.gov
20250829-NatzCeremonyVoterRegistration	https://www.uscis.gov/sites/default/files/document/policy-manual-updates/20250829-NatzCeremonyVoterRegistration.pdf	eac.gov, govinfo.gov, gpo.gov, house.gov, uscis.gov, vote.gov
20250829-VoterRegistrationGMC	https://www.uscis.gov/sites/default/files/document/policy-manual-updates/20250829-VoterRegistrationGMC.pdf	ecfr.gov, govinfo.gov, house.gov, uscis.gov
20250926-MilitaryNaturalization	https://www.uscis.gov/sites/default/files/document/policy-manual-updates/20250926-MilitaryNaturalization.pdf	govinfo.gov, house.gov, uscis.gov, whs.mil

Table 5: Data sources for every task in AnalystBench. *Note:* Full URLs and reconstruction scripts for all tasks will be released following legal review. Tasks drawing exclusively from U.S. government sources are available immediately.

A.2 End-to-end description of two `AnalystBench` tasks

We show example report generation requirements and workflows for two example tasks: `columbus` (2024 variant, which is synthesis-oriented; Figure 3) and `acts` (2025 variant, which is summary-oriented; Figure 4).

A.3 Obtaining reference output documents

➤ **Brainstorming with LLMs:** Using LLMs as ideation tools (Li et al., 2025b), we prompt GPT-5 with web search tool to propose task candidates that satisfy our desiderata for each domain (Figure 18), which are then manually reviewed by the authors. After roughly three to four calls per domain, the model's suggestions converge in multiple duplications. We obtain 14 tasks via this approach.
➤ **Programmatic web search:** We use SerpAPI[15] to search for tasks with multiple years or categories of variants. Our queries that combine task-type keywords (e.g., "report", "summary") with explicit year ranges ("2025 2024 2023 2022 2021"). We obtain 6 tasks from this approach.

A.4 Task validity

Figure 5 shows the average coverage rate by task. Figure 6 shows the percentage of irrelevant links included per task. Table 6 shows the error types for the ungrounded claims.

B API setup

Table 7 shows the setup details used for LLMs and search APIs in this paper.

C Expert evaluation on the validity of tasks, checklists, and LLM judgments.

C.1 Setup

We recruit experts from the Upwork platform.[16] The experts are vetted by Upwork recruiters, get paid $25 an hour, 3 hours per instance. Figure 7, Figure 8, Figure 9, Figure 10, and Figure 11 show screenshots of the annotation interface, which is hosted with LabelStudio.[17] 3 annotators annotate the same instances for each domain, one variant from each task to establish agreement. In total, we have 12 experts over 4 domains. Each of the 20 checklists is evaluated by 3 annotators across four task domains: *Economic/Finance analysis*, *Legislative research*, *Public policy/Governance*, and *Statistical analysis*. We clearly inform experts that their data and feedback are being collected to build a benchmark and may be used in a research publication.

C.2 Agreement analysis

Annotator agreement is measured across four dimensions: task feasibility, checklist clarity, checklist comprehensiveness, and checklist suitability for evaluating the task (Table 8). Overall, annotators agreed most strongly on task feasibility (0.867 pairwise agreement and 0.963 Gwet AC2). In contrast, agreement drops for checklist quality, including comprehensiveness (pairwise 0.433; Gwet AC2 0.706) and suitability (pairwise 0.333; Gwet AC2 0.594), which is also reflected in the larger spread of rater-consensus deltas in (Figure 12). The seemingly low Krippendorff's α is consistent with this setup: label distributions are highly skewed toward "yes," sample sizes are small, and some raters seem to be stricter or looser, all of which can result in low α even when majority agreement and Gwet's AC2 indicate meaningful reliability.

C.3 Checklist feedback

Table 9 shows the most common feedback for each evaluation dimension.

C.4 LLM judgment feedback

In the initial experiment, we use GPT-4.1 without scoring guidelines, and thus human experts find 82 criteria with evaluation issues (on average 1.36 problematic criteria out of 15 per task). The errors either parsing errors (e.g., a score of 1/5 being parsed as 0/5) or cases where the assigned scores are more lenient than expert judgment. Based on these findings, we upgrade the judge to GPT-5.1, fix parsing logic, and introduce explicit scoring guidelines. The experts do not find any issue after these changes.

D Generating reports with LLMs

D.1 Error analysis

Table 10 summarizes execution errors that break both vanilla and agent generation. When these errors occur, we restart execution without modifying the configuration, except for context-length errors where we use a stricter token limit. The most common issue is LLM API rate limiting (29.9%),

[15] https://serpapi.com/
[16] https://www.upwork.com/
[17] https://labelstud.io/

Task 1: Columbus Popular Annual Financial Report (`columbus_2024`)

```
# Task goal:

Write an accessible Popular Annual Financial Report for the City of Columbus (Dec. 31, 2024)
↪  based on the annual comprehensive financial report, budget materials, council/government
↪  docs, and related sources.

## Inputs:

33 documents multimodal city finance documents, including annual comprehensive financial reports,
↪  budget materials, council/government docs, and related sources.

## Expected output:

A structured report with sections that cover economy, revenues/expenditures, assets, debt,
↪  projections, including financial figures, tables that are a subset of those from the input
↪  documents, and illustrative visualizations.

# What makes it hard:

The model will have to process 33 multimodal city finance documents into an accessible report of
↪  around 10 pages, with exact dollar figures aggregated from the correct source tables and
↪  charts. Specifically, the model will have to:

1. Find the right facts across many files and keep them consistent.
2. Ensure quantitative precision, especially for a financial report.
3. Explain technical accounting terms clearly.
4. Generate charts/tables that match the underlying data.
```

Figure 3: Task 1 – Columbus Popular Annual Financial Report (`columbus_2024`)

which appears as timeouts or throttling when processing large volumes of input documents. The second most frequent errors are agent generation failures (10.1%), which occur when an agent terminates without producing a usable output. All other error types are rare. Context-length errors (0.3%) occur when prompts exceed a model's context window, most notably for Qwen. Code-related issues, which includes missing dependencies (0.2%), missing input files (0.2%), and execution errors such as numerical or shape mismatches (0.4%), appear infrequently and are typically resolved by restarting the runs.

D.2 Stability analysis

We obtain a single generation for each configuration due to cost and time complexity. Here, we show that although repeated runs on a small subset show modest score variances, the core findings remain unchanged. To quickly quantify run-level stochasticity on a small scale, we re-run the OpenHands configuration on 15 instances (3 tasks – `kff`, `uscis`, `bls` – across all 5 variants each) for two additional times and score the outputs with GPT-5.1 using the original checklists. Results are summarized in Table 11. Both GPT-5.1 and DeepSeek-R1 show moderate variance in the mean scores (std of around 4–5 points), but GPT-5.1 outperforms DeepSeek-R1 by over 20 percentage points across every run. This indicates that the core findings are robust to run-level stochasticity.

E Prompts

Table 12 lists all the prompts used in our experiments.

Task 2: Connecticut Acts Affecting the Environment Summary (`acts_2025`)

```
# Task goal:

Write an accessible 10–15 page policy report summarizing 2025 Connecticut environmental
↪  legislation, based on the full text of enacted public acts and supporting reference materials.

## Inputs:

62 legal documents about public acts.

## Expected output:

A structured summary organized by policy area that (1) groups related acts together under clear
↪  headings, (2) summarizes each act's key provisions and "what changed", and (3)
↪  cross-references acts using exact IDs (e.g., "PA 25-58").

# What makes it hard:

The model will have to turn 62 separate legal acts into a coherent long-form report while
↪  maintaining legal precision. Specifically, the model will have to:

1. Find the right provisions across many acts and keep details (definitions, thresholds, dates,
↪  agencies, exemptions) consistent throughout the report.
2. Maintain identifier accuracy (public act numbers, internal cross-references) across summaries
↪  and cross-links.
3. Classify each act into the right policy area without missing edge cases where a bill touches
↪  multiple topics (e.g., energy + wetlands + permitting).
4. Connect separate acts into a single narrative.
5. Translate statutory language into plain-English "what changed" summaries without altering
↪  legal meaning.
```

Figure 4: Task 2 – Connecticut Acts Affecting the Environment Summary (`acts_2025`)

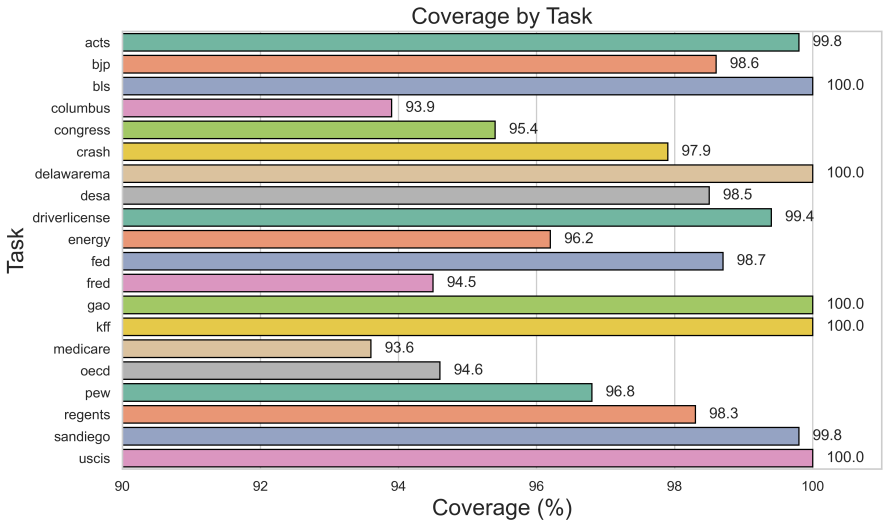

Figure 5: Coverage rate by task

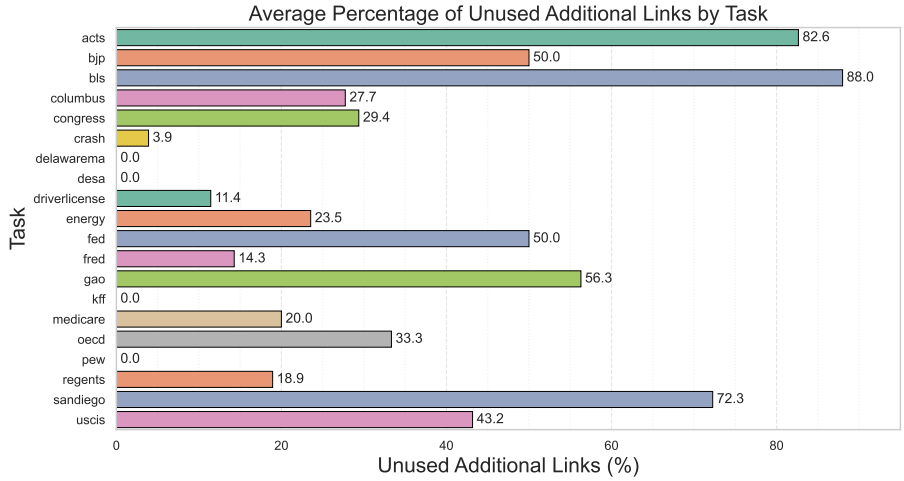

Figure 6: Average percentage of unused links by task

Error Type	**Count**	**Percentage (%)**
Unclear/Unclassified	80	51.3
Rankings or ordinal positions (e.g., “23rd”, “10th”)	16	10.3
Data from structured formats (HTML tables, CSV)	14	9.0
Complex multi-part claims (long sentences with multiple clauses)	12	7.7
Claims with pronouns/references requiring context (e.g., “this”, “those”)	10	6.4
Claims from parenthetical or explanatory text	9	5.8
Temporal comparisons requiring inference (e.g., “surpassed”, “fell below”)	5	3.2
Derived calculations (e.g., “per week”, “per day”)	3	1.9
Methodological or technical details	3	1.9
Percentage or ratio claims from comparisons	2	1.3
Claims from fragmentary or incomplete source text	2	1.3
Total	**156**	**100.0**

Table 6: Distribution of ungrounded claims by error type across tasks. Error types describe why claims extracted from output documents could not be grounded in source documents.

Model / Tool	Task	Hyperparameters	Cost / task (USD)	Cost / 100 variants (USD)
Data construction & automatic validation				
GPT-5	Brainstorming tasks	Default temperature, top_p, max_tokens; reasoning_effort=high; tool=web-search	$0.22 (≈3 tries/task)	$4.32
Serp API	Brainstorming tasks	Default parameters; top-10 results returned	$0.00 (free tier)	$0.00
GPT-5	Extracting references	Default temperature, top_p, max_tokens; reasoning_effort=high	$0.27 (2 calls: implicit & explicit)	$27.00
Serp API	Reference search	Default parameters; top-10 results returned	$0.00 (free tier)	$0.00
FireCrawl	Scraping documents	Default parameters	$0.16 ($16/month)	$16.00
GPT-4.1	Document relevance	temperature=0.0; max_tokens=32,768	$1.44 (30 calls/task)	$144.00
GPT-5	Search query refinement	Default temperature, top_p, max_tokens; reasoning_effort=high	$0.40	$40.00
GPT-5	Task instruction generation	Default temperature, top_p, max_tokens; reasoning_effort=high	$0.14	$13.50
GPT-5.1	Claim extraction (coverage Vaeck)	temperature=0.0; max_tokens=65,535	$0.14	$13.50
Section total: Data construction & auto validation			*$2.77*	*$258.32*
Vaecklist generation & evaluation				
GPT-5.1	Criteria generation	Default temperature, top_p, max_tokens; reasoning_effort=high	$0.07 ($10.20 over 100 instances)	$6.80
GPT-5.1	Evaluation	temperature=0.0; max_tokens=65,535	$0.56 ($84.00 over 100 instances)	$56.00
Section total: Vaecklist gen & eval			*$0.63*	*$62.80*
Report generation (100 tasks)				
GPT-5.1	ToC extraction	temperature=0.6; max_tokens=65,535	$1.52	$152.00
GPT-5.1	Generation	temperature=0.6; max_tokens=65,535; reasoning_effort=high	Va: $0.32 Ag: $0.47	Va: $31.60 Ag: $46.80
Gemini-3-Pro	Generation	temperature=0.6; max_tokens=65,535	Va: $0.57 Ag: $1.02	Va: $57.20 Ag: $101.60
DeepSeek-R1	Generation	temperature=0.6; max_tokens=65,535	Va: $0.04 Ag: $0.07	Va: $4.10 Ag: $7.40
Qwen3-32B	Generation	temperature=0.6; max_tokens=65,535; enable_thinking=True	Va: $0.01 Ag: $0.02	Va: $1.00 Ag: $1.90
Qwen3-14B	Generation	temperature=0.6; top_p=0.95; max_tokens=65,535; enable_thinking=True	Va: $0.01 Ag: $0.03	Va: $1.00 Ag: $3.00
Qwen3-8B	Generation	temperature=0.6; top_p=0.95; max_tokens=65,535; enable_thinking=True	Va: $0.01 Ag: $0.01	Va: $1.00 Ag: $1.00
Section total: Report gen, across models; incl. ToC			*Va: $1.53–$2.09* *Ag: $1.53–$2.54*	*Va: $153.00 –$209.20* *Ag: $153.00 –$253.60*

Table 7: Setup details for models/tools used across tasks. Costs are reported as *per task* and *per 100 tasks*. *Va* stands for vanilla generation and *Ag* stands for agent-based generation.

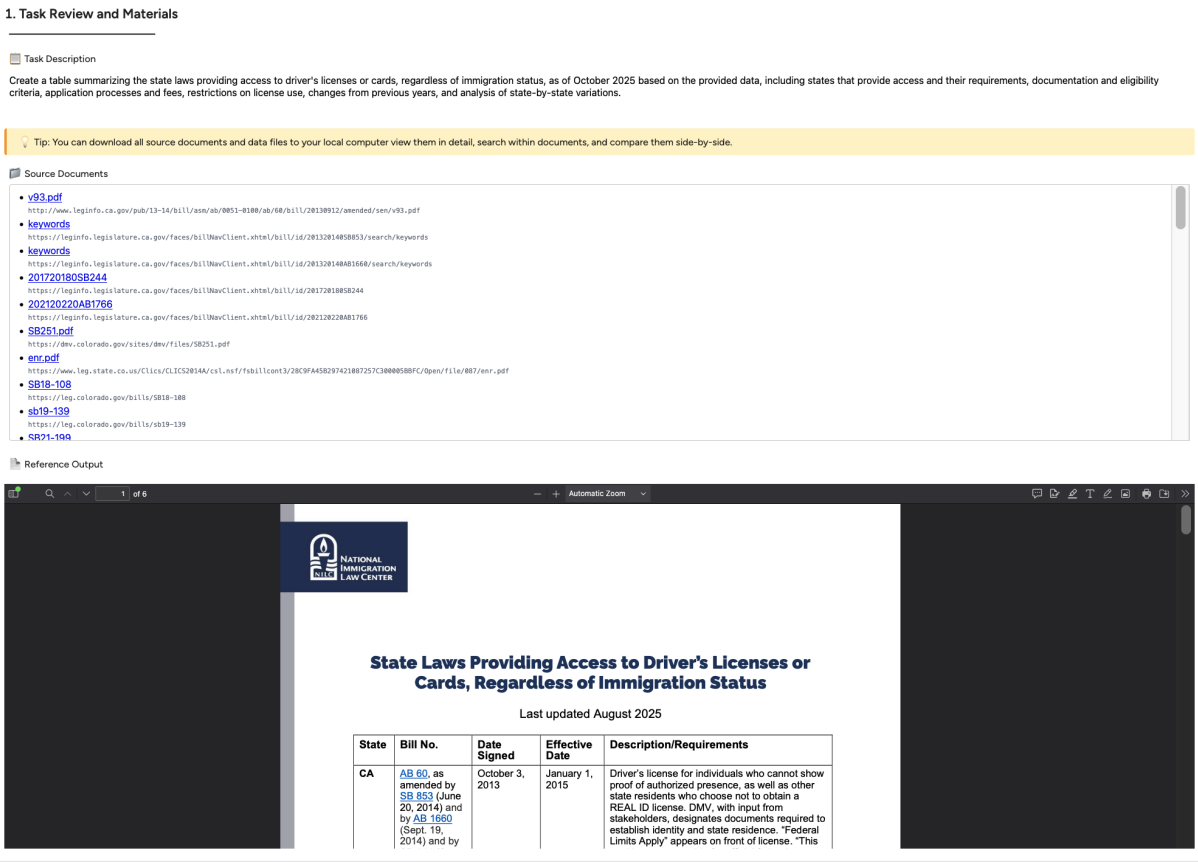

Figure 7: Part 1 of the annotation interface. The annotators read through task instruction, the source files, and the ground truth file to determine task validity.

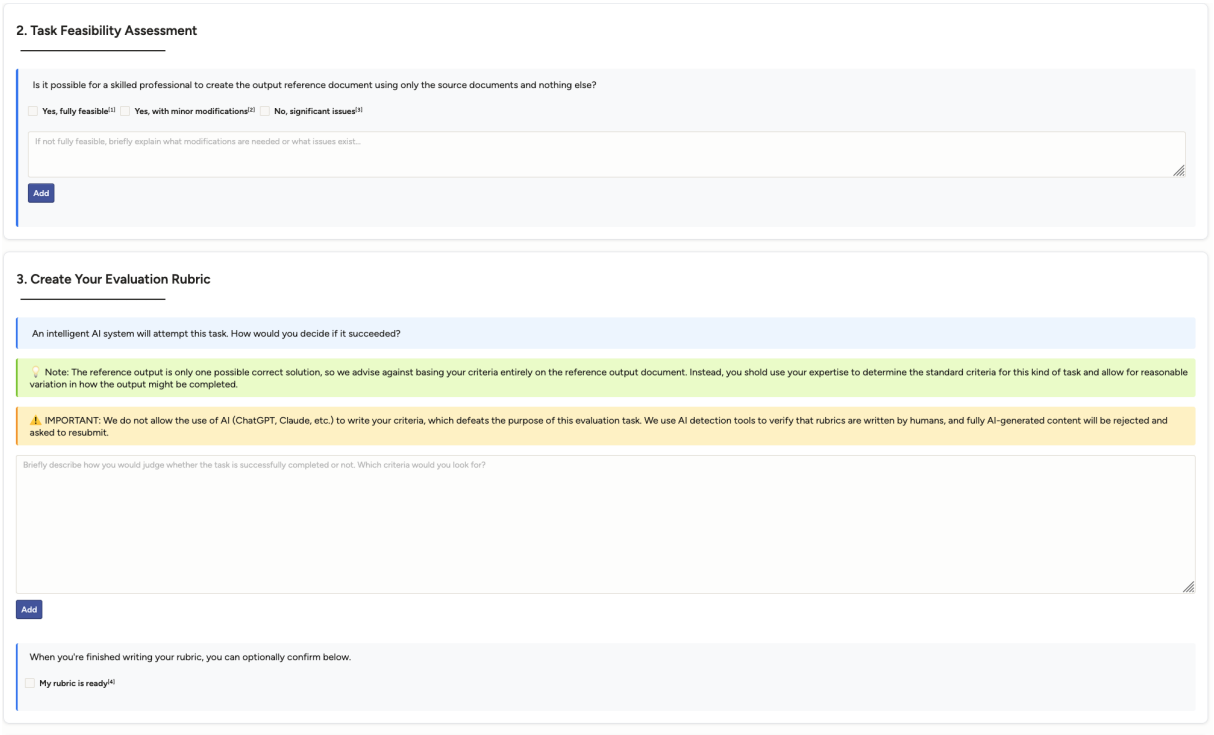

Figure 8: Part 2 of the annotation interface. The annotators give their judgments on the task feasibility and write their own rubric.

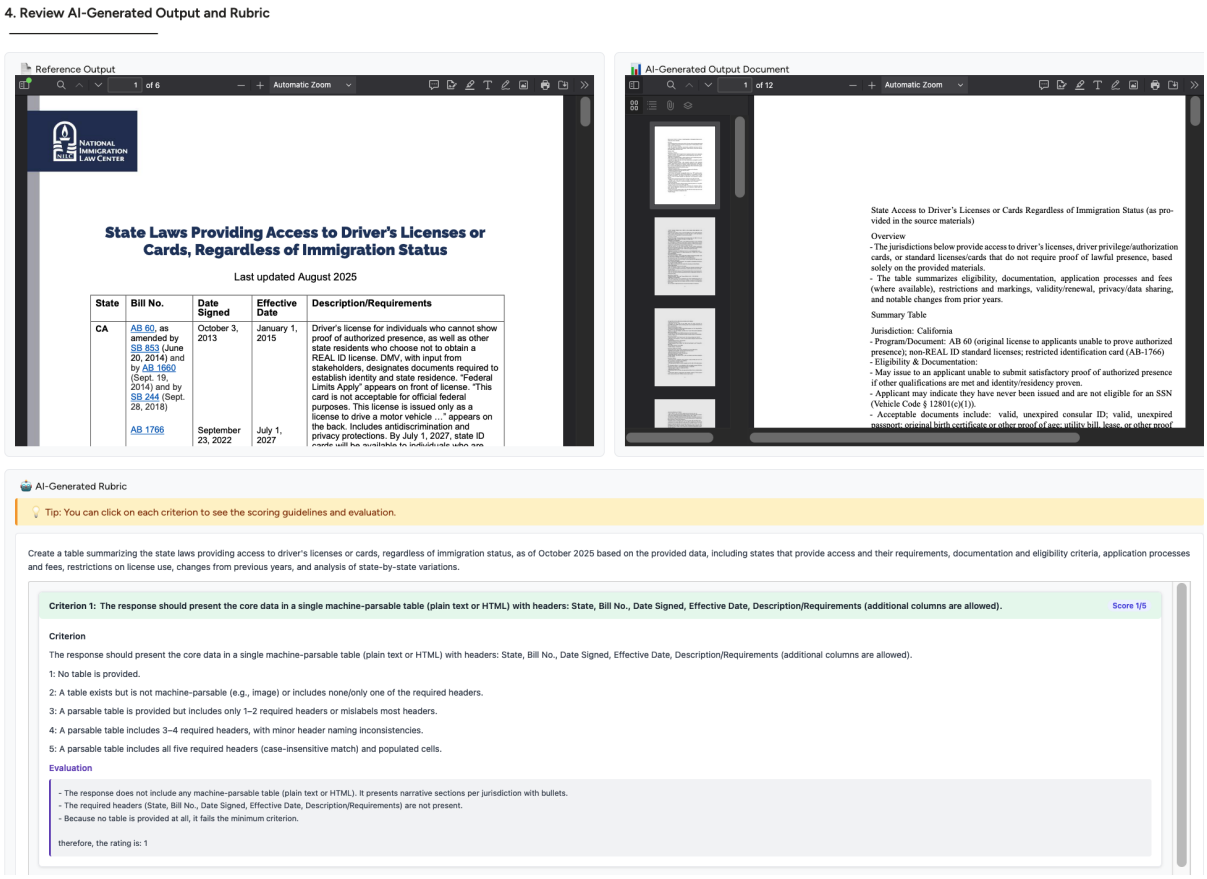

Figure 9: Part 3 of the annotation interface. The annotators review the AI-generated output and ground truth document, as well as the AI-generated checklist, which contains a list of criteria and the corresponding scoring guideline.

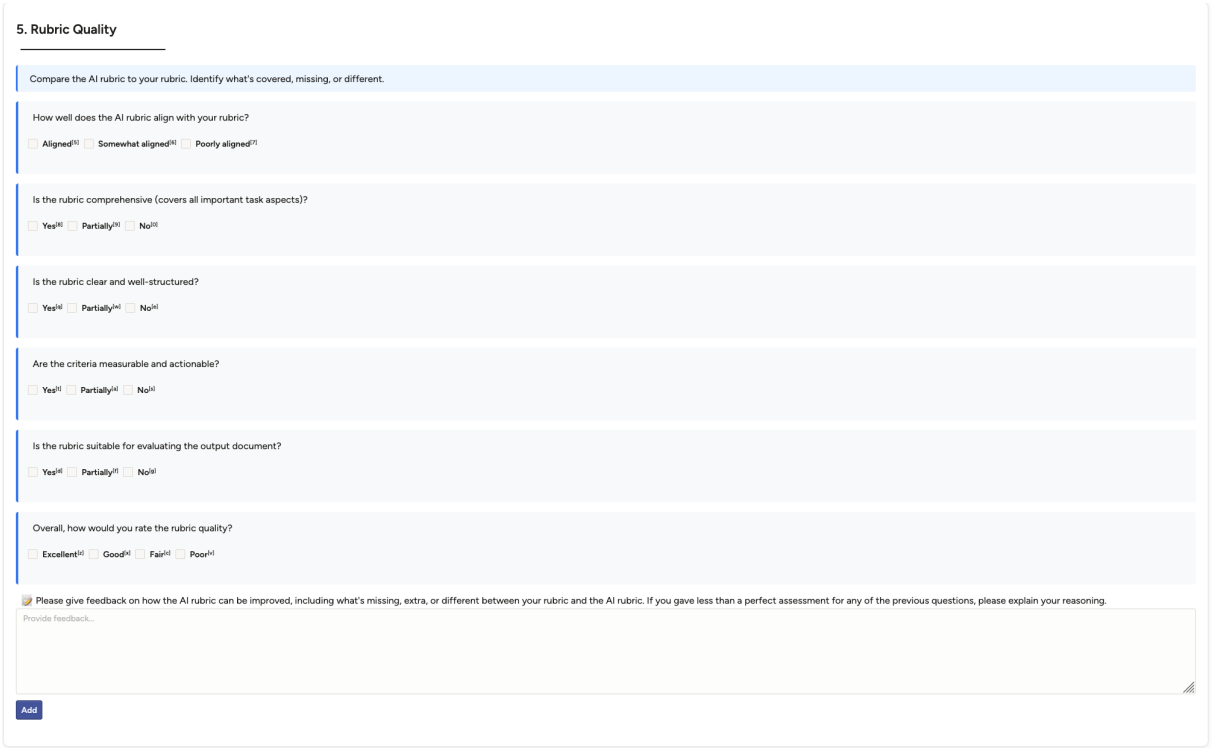

Figure 10: Part 4 of the annotation interface. The annotators give judgment on the checklist quality.

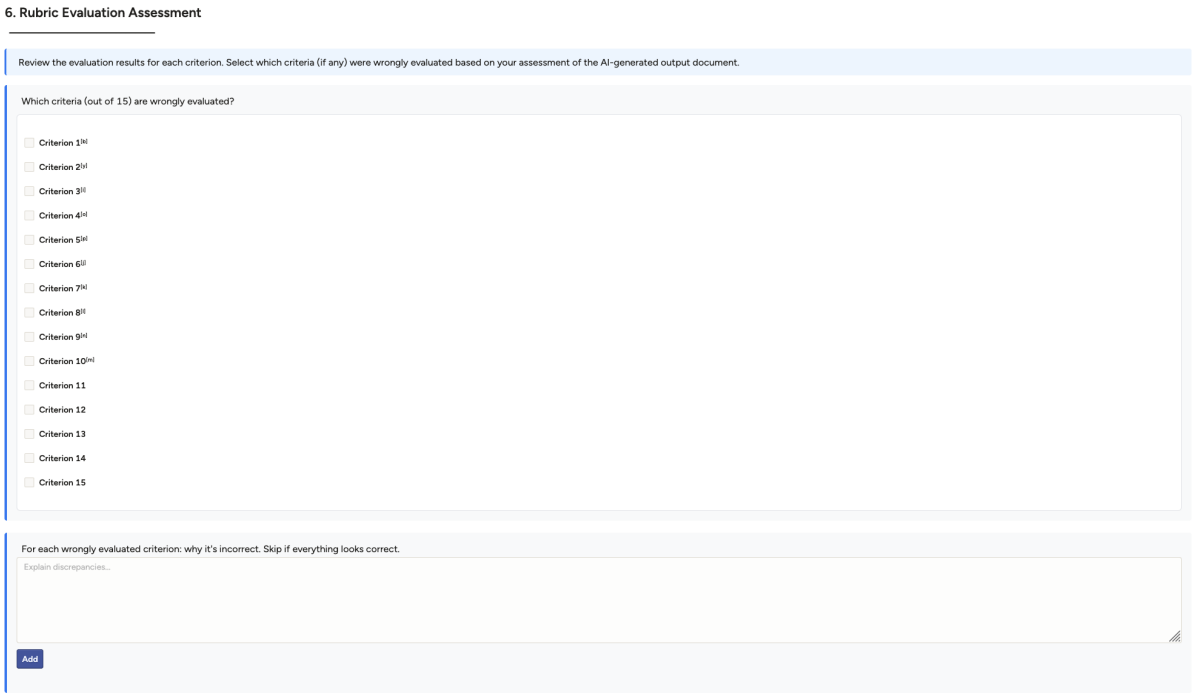

Figure 11: Part 5 of the annotation interface. The annotators choose which criteria in the checklist are wrongly evaluated and provide their reasoning.

Domain	Items	Unanimous (3/3)	Majority ($\geq$2/3)	Full Disagr.	Pairwise agreement	Gwet AC2 (ordinal)	Kripp. α (ordinal)	Label Dist. (no/part./yes)
Task feasibility								
Economic	6	0.833	1.000	0.000	0.889	0.971	0.000	0.00/0.06/0.94
Legislative	4	0.500	1.000	0.000	0.667	0.897	0.185	0.00/0.25/0.75
Policies	5	0.800	1.000	0.000	0.867	0.962	0.462	0.00/0.13/0.87
Statistics	5	1.000	1.000	0.000	1.000	NA	NA	0.00/0.00/1.00
Overall	**20**	**0.800**	**1.000**	**0.000**	**0.867**	**0.963**	**0.272**	**0.00/0.10/0.90**
Checklist clarity								
Economic	6	0.500	1.000	0.333	0.667	0.613	-0.133	0.17/0.00/0.83
Legislative	4	1.000	1.000	0.000	1.000	NA	NA	0.00/0.00/1.00
Policies	5	0.400	1.000	0.133	0.600	0.759	-0.135	0.07/0.13/0.80
Statistics	5	0.400	1.000	0.000	0.600	0.881	-0.167	0.00/0.20/0.80
Overall	**20**	**0.550**	**1.000**	**0.133**	**0.700**	**0.798**	**-0.136**	**0.07/0.08/0.85**
Checklist comprehensiveness								
Economic	6	0.000	0.667	0.333	0.222	0.354	-0.301	0.22/0.28/0.50
Legislative	4	0.750	1.000	0.000	0.833	0.955	0.000	0.00/0.08/0.92
Policies	5	0.200	1.000	0.000	0.467	0.825	-0.037	0.00/0.40/0.60
Statistics	5	0.000	1.000	0.000	0.333	0.778	-0.250	0.00/0.53/0.47
Overall	**20**	**0.200**	**0.900**	**0.100**	**0.433**	**0.706**	**-0.115**	**0.07/0.33/0.60**
Checklist suitability for evaluating the task								
Economic	6	0.000	0.167	0.389	0.056	0.212	-0.386	0.33/0.28/0.39
Legislative	4	0.500	1.000	0.000	0.667	0.903	-0.100	0.00/0.17/0.83
Policies	5	0.200	1.000	0.133	0.467	0.693	-0.167	0.07/0.27/0.67
Statistics	5	0.000	0.800	0.067	0.267	0.678	-0.140	0.07/0.40/0.53
Overall	**20**	**0.150**	**0.700**	**0.167**	**0.333**	**0.594**	**-0.116**	**0.13/0.28/0.58**

Table 8: Inter-annotator agreement for checklist quality evaluation.

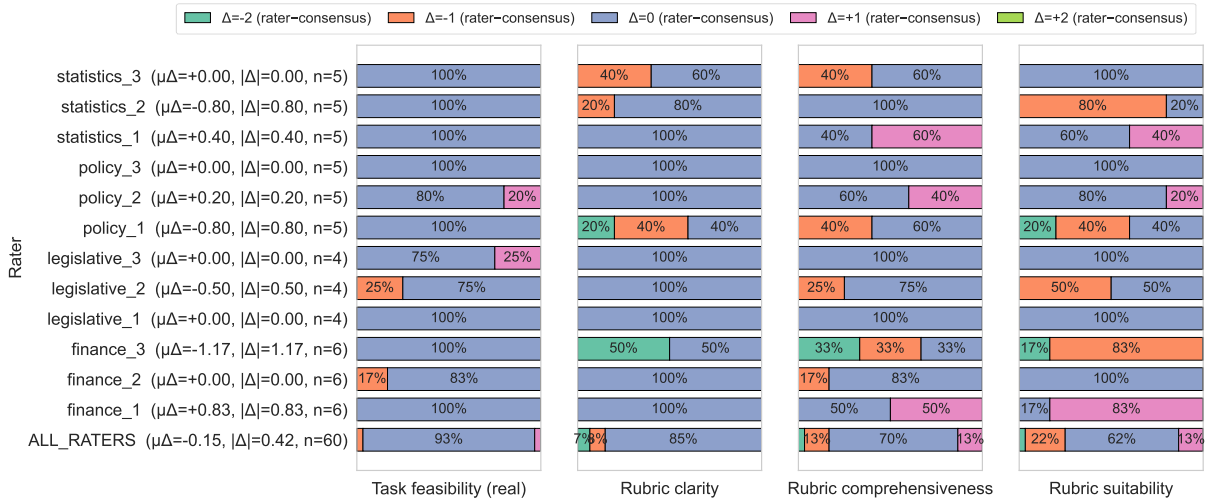

Figure 12: Rater disagreement across evaluation dimensions. Stacked bars show the distribution of severity deltas $\Delta \in \{-2, -1, 0, +1, +2\}$ between each rater's label and the consensus label for each domain.

Dimension	Positives feedback	Negative feedback
Clarity	• Clear and well-structured; easier to apply than free-form feedback • Makes expectations explicit (e.g., formatting, sectioning)	• Ambiguous wording; overlapping criteria • Missing definitions
Comprehensiveness	• Broad coverage of key quality dimensions (coverage, accuracy, structure) • Checks numeric/detail correctness beyond narrative content	• Encourages listing facts over synthesis or insight • Gaps for some tasks (e.g., time trends, demographics, domain nuances)
Alignment with expert rubrics	• Calls for more actionable language to improve annotator consistency • Some checklists are even more comprehensie and detailed than that of the experts	• Mismatch with expert priorities (substance vs. formatting)
Task suitability	• Occasionally useful for capturing policy or contextual framing	• Too strict on templates
Scoring & criteria	• Checklist-style criteria standardize scoring • Format/readability criteria useful for long reports	• Some overlapping criteria • More criteria on data presence than narrative quality

Table 9: Common feedback across rubric quality dimensions

Execution error	Description	Freq.	Models
LLM API rate limiting	Timeouts and rate-limit errors	29.9	
LLM API context length	Prompt length exceeds model context window	0.3	
Agent generation failure	Agent does not save a valid output	10.1	
Code dependency	Runtime failure caused by missing Python dependencies	0.2	
Missing input file	Generated code refers to missing data files	0.2	
Code execution mismatch	Numerical/dimensional errors in plotting/data operations	0.4	

Table 10: Common error types encountered during report generation, sorted by frequency (% of occurrences across all logs). Runs are restarted with minimal modifications after each error. Rate limiting errors are the most frequent due to the large amount of the input documents that need to be processed.

Model	Original mean	Rep1 mean	Rep2 mean	Std across 3 runs
GPT-5.1	87.9	78.9	82.4	4.53
DeepSeek-R1	60.0	52.9	60.0	4.11

Table 11: Run-level stochasticity analysis on a 15-instance subset (3 tasks × 5 variants) using the OpenHands configuration, scored by GPT-5.1 with the original checklists.

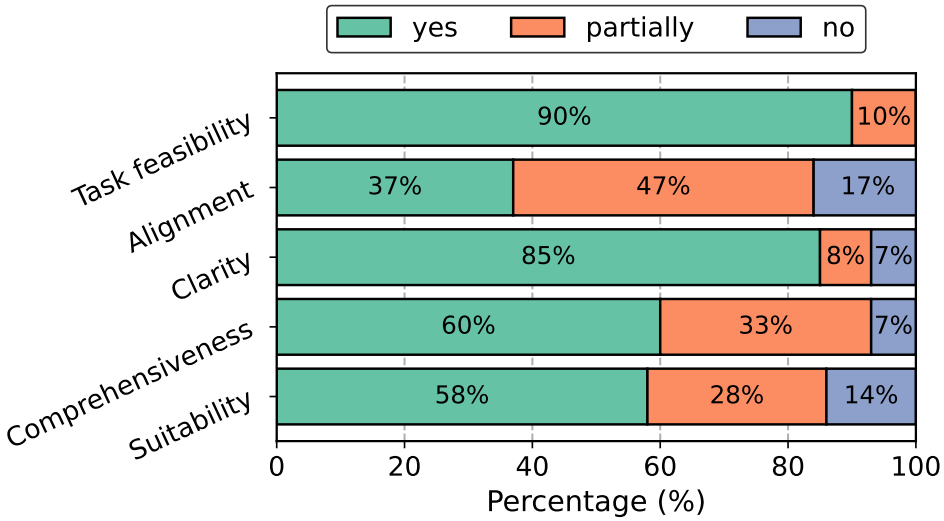

Figure 13: Expert ratings on rubric quality and task feasibility. Rubrics are judged as comprehensive and feasible, with lower agreement on alignment and task suitability.

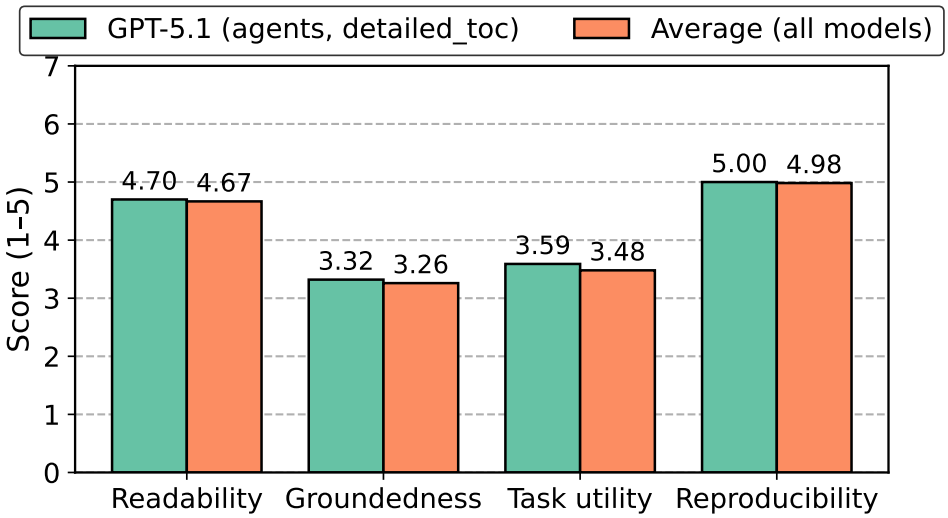

Figure 14: Score distribution for visual quality between the strongest setting (GPT-5.1, agents, ToC) and the average score across all settings. Reproducibility and readability are rated highly (near perfect for both settings).

Prompt	Caption	Reference
Checklist creation prompt	Full prompt used for checklist creation.	Figure 15
Checklist evaluation	Full prompt used for checklist evaluation (1–5 scale).	Figure 16
Claim extraction	Full prompt used for extracting verifiable claims from output documents.	Figure 17
Task collection	Full prompt used for collecting tasks with Deep Research.	Figure 18
Implicit references extraction	Full prompt used for extracting implicit references from reference output documents $\mathcal{R}$.	Figure 19
Explicit references extraction	Full prompt used for extracting explicit references from reference output documents $\mathcal{R}$.	Figure 20

Table 12: Summary of prompts used in our experiments

Checklist Creation Prompt

```
Your task is to create a detailed and comprehensive evaluation checklist for judging an AI
↪  assistant's response. You will be given an instruction and a reference output document that
↪  serves as the gold standard.

## Instruction:
{instruction}

## Reference Output Document:
{ref_content}

{exclusion_text}

## Checklist Generation Guidelines:
Based on the reference document, generate a comprehensive checklist of {n_criteria} criteria.
↪  Each criterion must be a specific, objective statement about what the response should contain
↪  or how it should be formatted.
1. Each criterion must be independently understandable and verifiable without access to the
↪  reference document. DO NOT refer to the reference document directly.
2. Each criterion must specify one requirement only (no compound checks).
3. Each criterion must include a 1–5 scoring scale, with five concise bullet points describing
↪  what qualifies as a 1, 2, 3, 4, or 5.
4. Global criteria that evaluates the entire report as a whole must be included:
    - Include at least one criterion that requires the response to include all key sections in the
    ↪  reference document, and list the names of all key sections explicitly.
    - Include at least one criterion that evaluates whether the response fully and directly
    ↪  answers the given instruction.
    - Include at least one criterion that checks if the response is overall coherent and readable,
    ↪  according to the domain standard of the task.
    – Include at least one criterion that evaluates whether the tone and level of jargon are
    ↪  appropriate for the task's domain.
5. All remaining criteria must cover essential informational or structural requirements
↪  demonstrated in the reference document:
    - For criteria about data, mention the specific stats, keywords, or key points.
    - For criteria about visual elements, describe the elements fully (e.g., "a bar chart with
    ↪  'X' and 'Y' axes showing values A, B, C" not "the chart").
    - Each criteria should require the response to use the correct format for presenting
    ↪  information (e.g., tables, lists). If the reference uses a 3-column table for data, a
    ↪  criterion must check for a 3-column table with the correct headers and data, not just the
    ↪  data presented in a list. A format mismatch should be penalized.
6. Formatting of all criteria must be consistent and follow the structure specified in the Output
↪  Format section below.
7. Double check to make sure that the criteria do not duplicate one another.

## Output Format:
Provide a list of {n_criteria} criteria, each formatted exactly as follows:

- The response should... (criteria description here)
    1: (scoring description here)
    2: (scoring description here)
    3: (scoring description here)
    4: (scoring description here)
    5: (scoring description here)

- The response should... (criteria description here)
    1: (scoring description here)
    2: (scoring description here)
    3: (scoring description here)
    4: (scoring description here)
    5: (scoring description here)

(continue until all {n_criteria} criteria are listed)
```

Figure 15: Full prompt used for checklist creation.

Checklist Evaluation Prompt

```
You are given a professional instruction and a professional's response to it. Your job is to
↪  evaluate how well the response meets the specific criteria provided.

When evaluating, focus only on:
        1.        The professional's response (including any Python code used to create plots or
        ↪  visualizations).
        2.        The evaluation criteria listed below.

Do not assume additional requirements or introduce your own expectations beyond what is
↪  explicitly stated in the criteria.

Instruction given to the professional:
{instruction}

Response from the professional:
{response}

Evaluation criteria:
{criteria}

Carefully review the response step by step. Then rate how well it satisfies the criterion using
↪  the provided 1--5 scoring guideline.

Think step by step and end your reasoning with "therefore, the rating is: X", where X is one of:
↪  1, 2, 3, 4, or 5.
```

Figure 16: Full prompt used for checklist evaluation (1–5 scale).

Claim Extraction Prompt

```
You will extract *atomic factual claims* from the document below.

Follow these rules:
- Break down every complex sentence into the smallest possible factual statements that can each
↪  be verified independently.
- Exclude any claims that are:
    (1) Commonsense facts anyone could know without the document (e.g., "California is a state in
    ↪  the USA").
    (2) Introductory statements that simply set up or summarize the document.
    (3) Subjective statements or personal opinions.
    (4) Metadata about the document itself (e.g., author, date, boilerplate text) that does not
    ↪  add to its core content.
- Return ONLY a single JSON object where:
  - each KEY is a claim string
  - each VALUE is the exact quote text copied verbatim from the document
- Do not include any explanations or extra fields.
- If the claim contains a statistics or entity (year, location, org), please include that exact
↪  statistics or entity in the claim. DO NOT round the numbers.

Task:
{task}

Document:
{doc}

Return the claims in the following json object:
{{
    "Atomic claim 1 here": "Corresponding original quote from which the claim is extracted here",
    "Atomic claim 2 here": "Corresponding original quote from which the claim is extracted here"
}}
```

Figure 17: Full prompt used for extracting verifiable claims from output documents

Task Collection Prompt

```
Construct a specific, realistic workplace subtask under the provided high-level task where a
↪  {job_title} processes multiple documents to create a single, detailed output document.

The subtask must follow this three-step workflow:
1. Gather and review several input documents that are publicly available on the web.
2. Extract and combine relevant information
3. Produce a single output document, where each claim is grounded in the input documents. The
↪  output document should ideally be shorter than 10 pages.

IMPORTANT: The output document must be mostly grounded in the input documents that are publicly
↪  available on the web. Please avoid using output documents with proprietary information that
↪  cannot be found on the web. All tasks without a publicly available output document or a
↪  document that is grounded mostly in proprietary information will be rejected.

## High-level task
{broad_task}

## Response format
Task: [Brief description of the specific subtask]
Input documents:
[List links to required input documents. Each link must lead to an existing document.]
Output document:
[Link to the expected output document. The link must lead to an existing document. If the output
↪  content is embedded within the linked document, please include the exact page range to
↪  extract the output content.]
(You may provide more than one subtask if applicable. Separate consecutive subtasks using "---"
↪  on a new line)

Remember, the subtask should focus on processing existing, publicly available documents to
↪  produce a single, synthesized output document that already exists on the web. Do not generate
↪  new content or rely on sources that are copyrighted or inaccessible. Provide direct, publicly
↪  accessible links to all required documents.
```

Figure 18: Full prompt used for collecting tasks with Deep Research

Implicit References Extraction Prompt

```
Your task is to identify any sentences in the provided document that require further evidence
↪  from external documents. You should follow these steps:

1. Read the document carefully.
2. Entity harvest: Find every unique entity that appears (e.g., organization, person, programs,
↪  etc.).
3. Statement mapping: Flag every sentence that
    (a) contains a quantitative value, citation, or statutory claim OR
    (b) describes the actions of any entity from Step 2.
    - Tip: These sentences should not be supported by any information in the same provided
    ↪  document.
    - Tip: If the sentences are too long, please break them down into more concrete claims.
    - Tip: Only flag obviously unsupported sentences.
4. Search query formulation: Formulate a concise search query to retrieve the supporting
↪  documents for each flagged sentence.
    - This query will be used to retrieve the most relevant supporting document via a Google
    ↪  Search.
    - This query should incorporate key information about the flagged sentence and the provided
    ↪  document (including unique entities from the document)
    - Keep the query INFORMATIVE but CONCISE to maximize search results. At most 2 verbatim
    ↪  keywords should be included, and the whole query should not exceed 10 tokens.
    - Tip: If the flagged sentence includes a statistics and/or entity (year, location, org), try
    ↪  to include that exact statistics and/or entity in the query, wrapped by quotation marks.
    - Tip: If the provided document is about a specific entity (year, location, org), try to
    ↪  include that exact entity in the query.
5. Compile your sentences into a numbered list. Each entry should follow this format:
    [Number]. [Flagged document sentence]: <search_query>[Concise, informative
    ↪  query]</search_query>

The response is incomplete unless every flagged sentence appears in the list. Only return the
↪  final numbered list. Keep the search queries informative but short.
```

Figure 19: Full prompt used for extracting implicit references from reference output documents $\mathcal{R}$

Explicit References Extraction Prompt

```
Your task is to identify references that are not already paired with a link anywhere in the
↪  provided document. You should follow these steps:

1. Read the document carefully.
2. Flag any reference that:
    - Appears to be a citation or document reference
    - Has NO link provided anywhere in the document (not immediately after, not later in the text)
4. Create search queries for each flagged citation:
    - This query will be used to retrieve the most relevant supporting document via a Google
    ↪  Search.
    - Tip: If the citation is too ambiguous, then add 1-2 keyword from the provided document to
    ↪  the query.
    - Tip: leave the citation text as is if the citation is not ambiguous.
5. Compile your sentences into a numbered list. Each entry should follow this format:
    [Number]. [Flagged document citation]: <search_query>[Concise, informative
    ↪  query]</search_query>

The response is incomplete unless every flagged citation appears in the list. Only return the
↪  final numbered list. Keep the search queries informative but short.
```

Figure 20: Full prompt used for extracting explicit references from reference output documents $\mathcal{R}$

Visual Quality Evaluation Prompt

```
You are given a task and two figures: a baseline figure from a reference document and a generated
↪  figure from a model's report.

Your job is to score the generated figure on a 1-5 scale across four dimensions, using the
↪  baseline figure and the task as grounding. Focus only on what is visible in the figures and
↪  any provided render status.

Dimensions (score 1-5 each):
1. Readability: chart type clarity, labels, titles, axes, legend, theme, and visual legibility.
2. Groundedness: variable/field match (entities, units, categories), trend/relationship match
↪  (directionality, relative ordering), quantitative plausibility (within tolerance if values
↪  visible), and chart appropriateness for the message.
3. Task utility: whether the figure helps advance or elaborate key insights for the task
↪  (relevance and explanatory power).
4. Reproducibility: whether the figure rendered successfully and is not missing. If the generated
↪  figure is missing or failed to render, this must be 1.

Return ONLY valid JSON with this schema:
{
  "readability": {"score": 1-5, "reason": "..."},
  "groundedness": {"score": 1-5, "reason": "..."},
  "task_utility": {"score": 1-5, "reason": "..."},
  "reproducibility": {"score": 1-5, "reason": "..."}
}

Be concise (1-3 sentences per reason). Do not include any text outside the JSON.
```

Figure 21: Full prompt used for judging the fine-grained quality of generated visualization