

DATA CROSS: A Unified Benchmark and Agent Framework for Cross-Modal Heterogeneous Data Analysis

Ruyi Qi^{1,*}, Zhou Liu^{1,*}, Yang Gu¹, Zeli Su¹, Zewei Pan¹

Xing Chen⁴, Yuanfeng Song⁴, Xiaoyue Wang¹, Wentao Zhang^{1,2,3,†}

¹Peking University ²Zhongguancun Academy

³Beijing Key Laboratory of Data Intelligence and Security

(Peking University) ⁴ByteDance

qrymaple@gmail.com, liuzhou82@gmail.com, gu_yang@pku.edu.cn

rickamorty@muc.edu.cn, pzp0057@sjtu.edu.cn, chenxing.xc@bytedance.com

songyf@outlook.com, 2501210103@stu.pku.edu.cn, wentao.zhang@pku.edu.cn

Abstract

Real-world analysis often requires combining directly queryable structured sources (e.g., SQL and CSV) with evidence preserved only in visual documents, such as scanned reports, invoices, and dashboards. Current analytics agents still focus primarily on structured and textual inputs, leaving this cross-modal setting underexplored. We introduce **DataCross**, a benchmark and agent framework for heterogeneous, insight-driven data analysis. **DataCrossBench**¹ contains 200 end-to-end tasks across six fine-grained categories: Fund Analysis, FinTech, Internet Development, Internet Firms, Short-form Drama, and AI Healthcare, grouped into four macro sectors and mixing SQL/CSV/JSON/TXT sources with image-based evidence. The benchmark is built with a human-in-the-loop reverse-synthesis pipeline to preserve cross-source dependency while keeping the target insights verifiable. We further propose **DataCrossAgent**, which coordinates source-specialized agents through intra-source exploration, source prioritization, cross-source pollination, and Reasoning ReAct (reReAct)-based executable verification. Experiments show that DataCrossBench reveals a difficulty gap for most baselines, especially on image-dependent Hard tasks. DataCrossAgent improves average factuality over direct GPT-4o prompting from 0.3291 to 0.4270 (+29.7%) and offers a competitive quality–cost profile in the profiled agent baselines. These results position DataCrossBench as a controlled, practical testbed for cross-modal heterogeneous data analysis.

1 Introduction

Large Language Models (LLMs) and multimodal foundation models have enabled data analysis

agents that can plan, write code, query data, and synthesize analytical narratives (Guo et al., 2025; OpenAI, 2025; Yang et al., 2025). Yet real analytical evidence is rarely contained in a single clean table. It is often fragmented across structured systems such as SQL/CSV/JSON files and visual-only artifacts such as scanned reports, invoice images, dashboards, and legacy screenshots (Wasserman et al., 2025; Chia et al., 2025). We refer to this operationally relevant but non-queryable visual evidence as *zombie data*.

Existing agents and benchmarks do not fully capture this setting. Many data-analysis benchmarks assume preprocessed structured inputs (Lai et al., 2023; Zhang et al., 2024; Li et al., 2023; Zhang et al., 2023; Majumder et al., 2025); multimodal QA and RAG benchmarks often emphasize document retrieval rather than executable, insight-driven analysis (Foroutan et al., 2025; Peng et al., 2025; Wasserman et al., 2025; Singh et al., 2025); and current protocols rarely test whether an agent can extract visual tables, align noisy entities or units with structured records, and produce verifiable cross-source conclusions (Sahu et al., 2025; Egg et al., 2025; Weng et al., 2026). This leaves a gap between benchmarked capabilities and the heterogeneous workflows encountered in practice.

We introduce **DataCrossBench**, a benchmark for unified, insight-driven analysis across heterogeneous sources. It contains 200 end-to-end tasks over six fine-grained categories, including Fund Analysis, FinTech, Internet Development, Internet Firms, Short-form Drama, and AI Healthcare. These categories are further grouped into four macro sectors. Tasks require evidence gathering across SQL/CSV/JSON/TXT files and, for the Hard split, visual documents containing charts or image-based tables. To keep the benchmark verifiable, we build it with a human-in-the-loop reverse-synthesis pipeline: expert-validated goals and target insights are used to synthesize the un-

*Equal contribution.

†Corresponding author.

¹**Benchmark.** Our dataset and code is available at <https://anonymous.4open.science/r/DataCrossBench/>.

derlying data artifacts, so that conclusions can be checked by execution.

On this benchmark, representative LMMs, specialized insight agents, and general-purpose CLI agents all face clear difficulty on vision-dependent Hard tasks, indicating that DataCrossBench fills an evaluation gap left by structured-data, data-discovery, and document-retrieval benchmarks (Majumder et al., 2025; Peng et al., 2025).

To further improve performance in this setting, we propose **DataCrossAgent**, a corresponding multi-agent framework for this benchmark. DataCrossAgent profiles each source, prioritizes high-value evidence, explores sources through Reasoning ReAct (reReAct), cross-pollinates hypotheses across sources, and verifies candidate findings with executable analysis before report generation. Its contribution is a task-specific, executable protocol that composes established agentic primitives; it is not intended as a new general-purpose reasoning primitive.

In a nutshell, our contributions are twofold:

- We introduce DataCrossBench, a 200-task benchmark that turns seed-report-grounded industrial scenarios into verifiable heterogeneous analytics tasks. It explicitly tests whether agents can recover evidence chains across SQL, CSV, JSON, text, and visual documents rather than solving isolated table or retrieval questions.
- We introduce DataCrossAgent, a source-specialized multi-agent framework that combines priority-guided routing, Reasoning ReAct-based executable exploration, cross-source hypothesis pollination, and final verification. Experiments show that this design improves factuality over vanilla GPT-4o and remains competitive with strong agentic baselines.

2 Related Work

Multimodal Document Reasoning. Multimodal document benchmarks study reasoning over tables, charts, PDFs, and medical reports. WikiMixQA and UniDoc-Bench focus on multimodal QA or RAG over long documents and PDF corpora (Foroutan et al., 2025; Peng et al., 2025), while MedInsightBench and Chart-RVR explore domain-specific insight discovery and verifiable chart reasoning (Zhu et al., 2025; Sinha et al., 2025; Nguyen and Okatani, 2026). These benchmarks demonstrate the importance of visual evidence extraction, but they generally emphasize retrieval or question

answering rather than executable, cross-source analytical workflows.

Data Analysis Benchmarks. Data-analysis benchmarks have progressed from semantic parsing and table QA to executable analysis over tabular datasets. DS-1000 and related evaluations emphasize Python/SQL tool use, debugging, and multi-step reasoning over structured data (Lai et al., 2023; Zhang et al., 2024; He et al., 2024; Hu et al., 2024). More recent insight-oriented benchmarks, such as InsightBench, AgentPoirot, DABStep, and UniDataBench, move beyond factoid answers toward higher-level analytical narratives (Sahu et al., 2025; Egg et al., 2025; Weng et al., 2026).

However, most of these settings still assume structured inputs or exclude visual documents, so they under-test analytics that must connect evidence across files and between structured and visual sources. To bridge this gap, we introduce our benchmark to specifically evaluate analytics that must seamlessly connect evidence across files and between structured and visual sources.

Agentic Data Analysis Systems. Data analysis agents combine LLM reasoning with tools for cleaning, exploration, modeling, visualization, and multimodal reasoning (Shen et al., 2025; Manatkar et al., 2025; Guo et al., 2024; Lu et al., 2026; Kumar et al., 2024; Sun et al., 2025). Yet existing agents often process sources sequentially or rely on shallow schema concatenation, creating a “silo effect” that weakens cross-system analysis over heterogeneous formats and granular logs (Just et al., 2024). DataCrossAgent targets this gap through source-specialized profiling, priority-guided routing, contextual cross-pollination, and executable verification over structured, textual, and visual evidence.

3 DataCrossBench: A Multi-Source Heterogeneous Benchmark

3.1 Design Philosophy and Comparison

Despite rapid progress in data analysis agents, existing benchmarks remain constrained by idealized assumptions about input format and modality. Classic datasets such as Spider focus primarily on Text-to-SQL, while DS-1000 extends evaluation to code generation but still relies on preprocessed files. These settings do not fully capture analytical workflows in which evidence is distributed across structured tables, semi-structured JSON/logs, and visual documents such as scanned reports or screen-

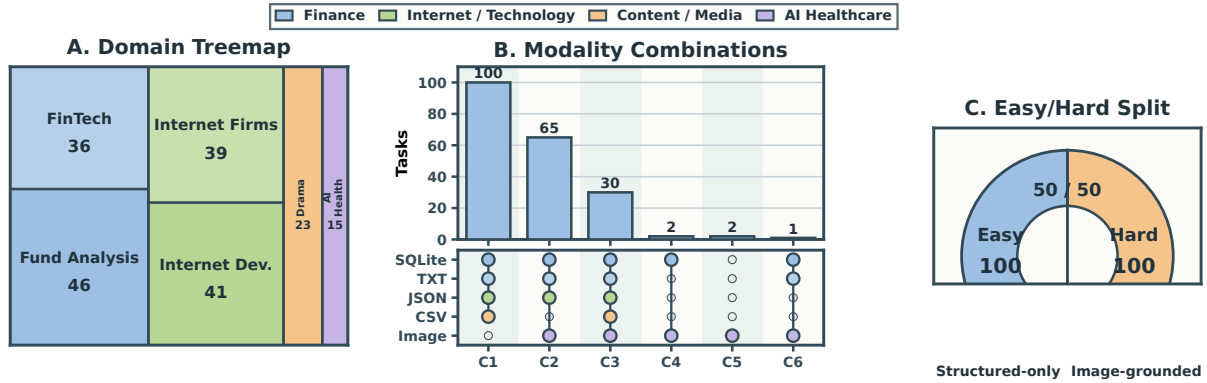


Figure 1: **DataCrossBench overview.** (A) Fine-grained category counts grouped by macro sector. (B) Frequent modality combinations across SQL/CSV/JSON/TXT/Image inputs. (C) Balanced Easy/Hard split, where Hard tasks require image-dependent evidence integration.

shots. In practice, useful conclusions often require aligning entities and units across files, integrating visual evidence with queryable records, and maintaining a consistent evidence chain under noise and redundancy. To address this gap, DataCrossBench evaluates end-to-end analysis in controlled, seed-report-grounded settings with executable verification. As shown in Table 1, it provides a multimodal, noisy, cross-file environment for testing whether agents can extract, align, and reason over heterogeneous evidence.

3.2 Data Construction

We employ a human-in-the-loop *reverse-synthesis* pipeline to ensure the data is both realistic and challenging. First, we curate authoritative reports (PDFs and industry write-ups) from six fine-grained topical categories: Fund Analysis, FinTech, Internet Development, Internet Firms, Short-form Drama, and AI Healthcare. These categories are grouped into four macro sectors: finance, internet/technology, content/media, and AI healthcare. We then utilize an LLM to extract high-level analytical goals and ground-truth insights, which are rigorously verified by human experts. Next, guided by these validated insights, we collaborate with a Code LLM to programmatically synthesize the underlying raw data files (e.g., SQLite databases, CSV tables, JSON metadata) required to derive the target conclusions. During this phase, we enforce complexity constraints on file count, modality mix, and fact-table scale, while still allowing smaller auxiliary dimensions and metadata files when cross-source alignment requires them. To prevent target leakage, this synthesis step excludes explicit answer strings, conclusion-like field names, and direct restatements of the ground-truth insights from

generated source files. Finally, the pipeline undergoes an iterative validation process: human annotators perform execution checks to identify and repair any consistency errors or crashes, followed by a definitive quality assurance review to guarantee that the synthesized multi-source data logically and accurately supports the ground-truth insights. After all structured data artifacts are completed, we produce auxiliary unstructured and multimodal resources as additional heterogeneous data sources, including accompanying .txt files and visualization charts. Although the released benchmark artifacts are synthesized for verifiability, the task objectives are not generated from unconstrained prompts. Instead, they are anchored in authoritative seed reports that expose quantitative evidence, domain context, and visually embedded tables or charts. We retain only source documents that satisfy four construction constraints: (i) **domain relevance**, so the task reflects a realistic analytical objective; (ii) **quantifiable insight**, so the target conclusion can be operationalized into executable checks; (iii) **cross-source recoverability**, so no single file trivially determines the answer; and (iv) **visual evidence availability**, so at least part of the evidence chain can be instantiated as image-based “zombie data.” Appendix Table 6 summarizes the report archetypes used during task design.

3.3 Dataset Statistics and Diversity

DataCrossBench comprises a total of 200 tasks, each designed as an end-to-end, multi-step data analysis problem requiring cross-source evidence gathering and reasoning. Based on the released benchmark metadata, the tasks span six fine-grained topical categories: Fund Analysis, FinTech, Internet Development, Internet Firms, Short-form

Table 1: Comparison of **DataCrossBench** with other benchmarks. We additionally highlight whether a benchmark involves **visual documents** (vision modality) that must be jointly analyzed with structured sources.

Dataset	Analysis Source	Multi-source	Multi-modal (Vision)	Task Size	File Size	Rows/File
<i>Specific Data Analysis</i> (Specific answer for a given question; e.g., “What is the average sales in Q3?”)						
DS-1000 (Lai et al., 2023)	StackOverflow	✗ (data.frame)	✗	1000	–	–
DSEval (Zhang et al., 2024)	LLM Gen. & Human	✗ (csv)	✗	825	44	23,630
Text2Analysis (He et al., 2024)	LLM Gen. & Human	✗ (csv)	✗	2249	347	–
DABench (Hu et al., 2024)	LLM Gen. & Human	✗ (csv)	✗	257	52	2,024
DABStep (Egg et al., 2025)	Real-world Data	✓ (3 types)	✗	450	7	3,121
<i>Multimodal QA / RAG</i> (Specific answer over visual documents; e.g., “According to this chart/PDF, what is the answer?”)						
WikiMixQA (Foroutan et al., 2025)	Wikipedia pages (tables & charts)	✓ (multi-page doc)	✓	1000	4000	–
UniDoc-Bench (Peng et al., 2025)	PDF corpus (page-level retrieval)	✓ (multi-doc/pages)	✓	1600	70000	–
<i>High-level Data Analysis</i> (Insights extracted for open-ended goals; e.g., “What latent causal mechanisms and non-obvious drivers underpin Q1 market expansion?”)						
InsightBench (Sahu et al., 2025)	LLM & Human	✗ (csv)	✗	100	112	545
UniDataBench (Weng et al., 2026)	Real-world Data	✓ (Multi-type)	✗	100	223	17,945
MedInsightBench (Zhu et al., 2025)	Medical cases (multimodal)	✗ (single case)	✓	332	–	–
DataCrossBench (Ours)	LLM & Human	✓ (4+ types)	✓	200	963	8,585

Drama, and AI Healthcare. We group them into four macro sectors: finance (Fund Analysis and FinTech; 82 tasks), internet/technology (Internet Development and Internet Firms; 80), content/media (Short-form Drama; 23), and AI healthcare (AI Healthcare; 15). The released inputs contain 963 declared files in total: 171 CSV files, 200 SQLite databases, 202 JSON files, 196 TXT briefs, and 194 images. Each task uses 4.82 input files on average (median 4, p90 8) and 4.09 modalities on average; 30 tasks exercise all five modalities. Following the benchmark protocol, we split the tasks evenly into Easy and Hard subsets according to whether image-based evidence is required: *Easy* tasks (100) involve only structured or textual sources (e.g., CSV/SQLite/JSON/TXT) and focus on cross-file reasoning without any image inputs; *Hard* tasks (100) are *image-dependent*: the target evidence chain requires parsing and integrating visual documents, introducing 194 images in total (1.94 per Hard task on average) and demanding chart/table interpretation, normalization, and cross-modal alignment with structured sources. Image presence alone does not define the split. Appendix Table 7 reports the full characterization summary used in this analysis.

3.4 Representative Task Patterns

To make the benchmark composition more concrete, Appendix Figure 6 visualizes a representative real task from the released benchmark. It illustrates the kinds of input sources, multimodal evidence, reasoning steps, and analytical outputs the benchmark requires while remaining concise enough to fit within the paper.

3.5 Quality Control and Human Verification

To ensure the robustness and accuracy of DataCrossBench, we implemented a rigorous two-stage quality control framework comprising automated sanity checks and expert human verification. Initially, an automated pipeline verifies the executability of all reference SQL and Python code and checks for data file corruption. Subsequently, we engaged annotators with professional backgrounds in data science to perform a double-blind review of the tasks. This manual verification process adheres to three strict criteria: (1) *Cross-source Necessity*, confirming that questions strictly require evidence gathering across multiple sources and cannot be answered by relying on a single file; (2) *Logic Chain Validation*, ensuring the reasoning path constitutes a unique and reasonable evidence chain; and (3) *Answer Consistency*, checking that the ground truth is consistent with manual calculations. To facilitate this workflow, we developed a dedicated human verification interface. Further details regarding the specific protocols and annotation guidelines are provided in supplementary material.

3.6 Evaluation Metrics

DataCrossBench tasks demand open-ended analytical reports and complex cross-source reasoning, which pose significant challenges for standard evaluation. While prior research has validated the feasibility of utilizing Large Language Models (LLMs) as evaluators for free-form analytical outputs, pure LLM-based scoring often suffers from **central tendency bias**, where the model leans toward assigning median scores and fails to capture nuanced quality differences. To mitigate this, we propose a comprehensive multi-dimensional scor-

ing framework that integrates deterministic heuristic metrics with LLM-based semantic assessment. This hybrid approach ensures a balanced evaluation across four key dimensions: **Factuality**, **Completeness**, **Logic**, and **Insightfulness**, thereby providing both the rigor of hard data verification and the depth of semantic understanding. The final score S_{total} is a weighted average of four key dimensions: **Factuality** ($w_f = 0.30$), **Completeness** ($w_c = 0.25$), **Logic** ($w_l = 0.20$), and **Insightfulness** ($w_i = 0.25$), where w_f is set a bit higher to emphasize numerical accuracy. In all result tables, **Overall** denotes S_{total} throughout the paper. This weighting intentionally prioritizes numerical correctness over stylistic fluency: a report with appealing language but incorrect figures is unusable in analytical practice. To quantify the stability of judge-dependent dimensions, we additionally report cross-judge consistency across GPT-4o, Gemini-2.5-Flash, and Qwen3-VL-32B-Instruct in Appendix H.

$$S_{total} = w_f \cdot S_{fact} + w_c \cdot S_{comp} + w_l \cdot S_{logic} + w_i \cdot S_{ins} \quad (1)$$

Factuality (S_{fact}). Factuality measures the alignment of numerical data and entity descriptions between the predicted insights (I_{pred}) and the ground truth (I_{gt}). We combine a deterministic numerical-consistency component with an LLM-based semantic factuality component. For numerical consistency, we extract all numerical values and units from the prediction and reference, and compute

$$S_{fact,num} = \frac{|N_{pred} \cap N_{gt}|}{|N_{gt}|}, \quad (2)$$

where N denotes the set of extracted numerical entities. For semantic factuality, $S_{fact,llm}$ is obtained from a customized LLM-judge prompt that assesses whether key entities, temporal trends, and directional claims are correct on a 1–10 scale. The final factuality score is $S_{fact} = 0.5 \cdot S_{fact,num} + 0.5 \cdot (S_{fact,llm}/10)$.

Completeness (S_{comp}). Completeness assesses the information coverage. We employ an embedding-based approach to measure the semantic overlap. For each ground-truth insight $g \in I_{gt}$, we find its maximum cosine similarity across all predicted insights $p \in I_{pred}$:

$$S_{comp} = \frac{1}{|I_{gt}|} \sum_{g \in I_{gt}} \max_{p \in I_{pred}} \cos(\mathbf{e}_g, \mathbf{e}_p) \quad (3)$$

where $\mathbf{e}$ represents the semantic embedding vector. **Logic** (S_{logic}). The logic dimension evaluates the internal coherence and reasoning quality. The LLM assesses: **Attribution**: validity of causal relationships; **Trend Inference**: evidence-based extrapolation; **Comparison**: objectivity of analysis. The score $L \in [1, 10]$ is normalized as $S_{logic} = L/10$. **Insightfulness** (S_{ins}). Insightfulness measures the value-add of the generated content. The LLM evaluates non-triviality, actionability, and novelty. To ensure stability, we adopt a G-Eval style approach, calculating a weighted average of the top-5 log-probabilities of the output tokens.

4 DataCrossAgent: A Corresponding Agent Framework

Formal problem. Given a realistic, task-specific heterogeneous source collection $\mathcal{S} = \{s_i\}_{i=1}^n$ and a natural-language analysis goal g , the system must generate an analytical report y that synthesizes findings across structured, semi-structured, textual, and visual evidence. Each report is evaluated against a verified reference insight set y^* using the weighted four-dimensional score $S_{total}(y, y^*, \mathcal{S})$ defined in Section 3.6. The objective is to maximize S_{total} while ensuring that each non-trivial claim can be grounded in evidence from the provided sources. Appendix A gives the complete task formulation, including source types, output schema, and cross-source alignment constraints.

DataCrossAgent addresses the fragmentation of cross-format data analysis through source-specialized agents coordinated by an explicit evidence hierarchy. As shown in Figure 2, the workflow first converts each source into a compact profile containing modality, schema or visual descrip-

Algorithm 1 Priority-guided DataCrossAgent Workflow

Require: Task goal g and heterogeneous sources $\mathcal{S} = \{s_i\}_{i=1}^n$
Ensure: Grounded analytical report y

- 1: **for all** source $s_i \in \mathcal{S}$ **do**
- 2: Build source profile p_i with modality, schema, entities, and candidate evidence.
- 3: Estimate $S_{obj}(s_i)$, $S_{sem}(s_i, g)$, and $S_{LLM}(s_i, g)$.
- 4: Compute $S_{priority}(s_i) = 0.4S_{obj}(s_i) + 0.3S_{sem}(s_i, g) + 0.3S_{LLM}(s_i, g)$.
- 5: **end for**
- 6: Rank sources by $S_{priority}$ and select top-ranked primary sources.
- 7: Run reReAct on primary sources and cross-pollinate hypotheses to auxiliary sources.
- 8: Verify joins or comparisons with executable code and synthesize y with supporting evidence.

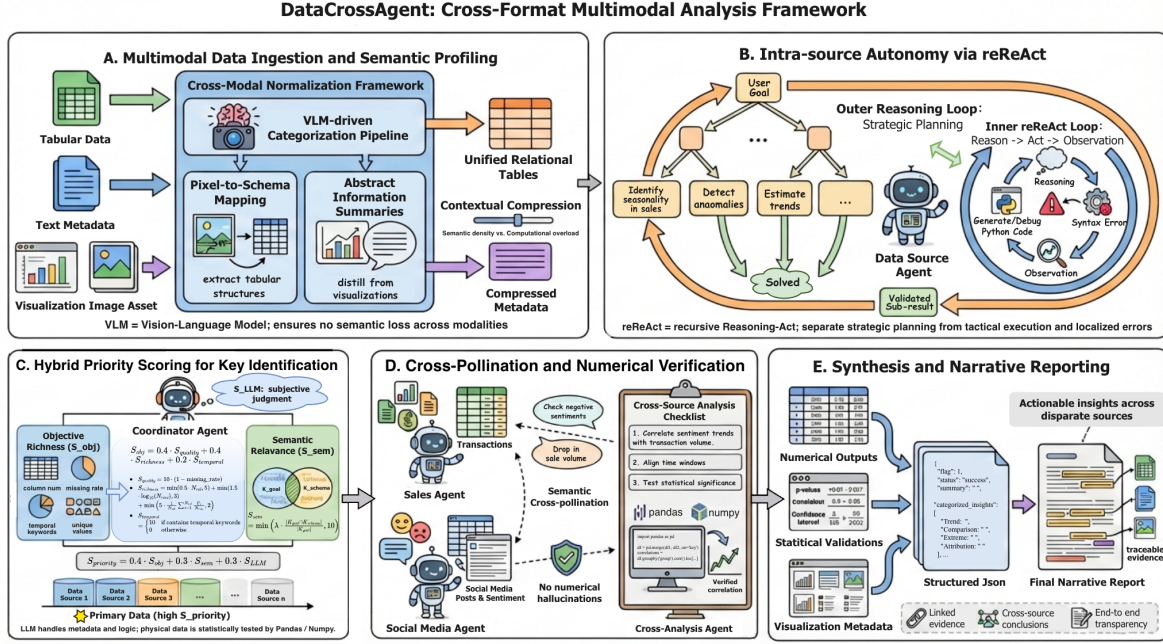


Figure 2: **The overall architecture of the DataCrossAgent framework.** A multi-agent orchestration to integrate heterogeneous data sources. The workflow first profiles each structured, textual, or visual source, then ranks sources through priority scoring before cross-source hypothesis exchange. Candidate insights are finally checked through executable analysis and evidence grounding before the report is generated.

tion, candidate entities, and potential evidence anchors. The coordinator then estimates objective quality, semantic relevance, and coordinator utility for each source, aggregates them into $S_{priority}$, and ranks sources before cross-source reasoning; full formulas are provided in Appendix G.

Algorithm 1 summarizes the execution path used throughout our experiments. The key design choice is to make source selection explicit before cross-source exploration: DataCrossAgent first profiles and ranks all sources, then applies reReAct to the selected primary sources, propagates hypotheses to auxiliary sources, and verifies candidate findings with executable code before final synthesis.

4.1 Source Profiling and Prioritization

DataCrossAgent normalizes heterogeneous inputs into source profiles rather than forcing all evidence into a single table. Structured files expose schemas and executable fields; textual files contribute compressed summaries and named entities; visual inputs are converted into table-like records or chart descriptions depending on asset type. This profiling stage gives the coordinator a comparable view over SQL/CSV/JSON/TXT/image sources, after which high-priority sources are selected as primary pivots and lower-ranked sources are retained as auxiliary evidence.

4.2 reReAct and Cross-Pollination

Each selected source is explored through a Reasoning ReAct (reReAct) loop that separates strategic decomposition from local executable debugging. Once source-level findings are obtained, DataCrossAgent performs cross-pollination: hypotheses from primary sources are projected onto auxiliary sources to search for shared entities, time windows, metrics, or contradictory evidence. This design keeps exploration focused while still allowing heterogeneous sources to challenge or support one another. In practice, the reReAct loop maintains a lightweight state for each source, including attempted queries, intermediate errors, extracted entities, and candidate findings. This prevents the coordinator from repeatedly issuing equivalent operations after a failed query or visual extraction step. Cross-pollination is then applied as hypothesis transfer rather than unrestricted conversation among agents: a finding from one source is converted into a compact query target and checked against other sources that share compatible entities, dates, or numerical fields. If an auxiliary source contradicts the hypothesis, the finding is revised or withheld.

4.3 Executable Verification and Synthesis

Before a cross-source hypothesis is promoted to the final report, DataCrossAgent resolves naming or

unit mismatches and verifies joins, comparisons, or aggregates with executable Python code. The final synthesis step then merges verified numerical outputs, source summaries, and visual evidence into a grounded analytical report. This execution-backed synthesis is intended to reduce numerical hallucination while preserving an auditable evidence trail for the generated insights.

5 Experiments

5.1 Experimental Setup

Baselines. We compare DataCrossAgent with three groups of baselines. i) **Large Multimodal Models (LMMs).** We directly use several LMMs to generate insights, including GPT-4o (OpenAI, 2024), Qwen3-vl (Bai et al., 2025), and Gemini-2.5-flash (Google, 2025). ii) **Specialized Agent Frameworks.** Given a dataset and an objective, these frameworks (AgentPoirot (Sahu et al., 2025) and ReActInsight (Weng et al., 2026)) perform end-to-end insight mining through continuous questioning. In our experiments, we organize heterogeneous inputs into usable tabular formats and allow each agent to iteratively generate answers. iii) **General-Purpose CLI Agent Systems.** We additionally evaluate Codex CLI and Claude CLI as strong off-the-shelf agentic baselines. These systems support iterative planning, tool use, code execution, and external-file inspection in a command-line loop, making them practically relevant baselines for multi-step heterogeneous analysis even though they are not customized for our benchmark. The Table 2 comparison uses one shared final-report interface for every system: a five-field JSON schema (flag_id, goal, used_files, insights, and summary) and the generic operators trend, comparison, extreme, and attribution. These fields specify report structure rather than instance answers, and the same instruction was supplied to all baselines.

Agent deployment details. In each agent framework, we use GPT-4o as the backbone LMM. All LMMs are configured with temperature 0 to ensure deterministic outputs. During operation, agents are set to explore questions for 2 iterative rounds, generating 2 new questions per round to ensure comprehensiveness. The Table 2 run used codex-cli 0.121.0 and Claude Code 2.1.107; all 200 tasks for each CLI baseline used one consistent version. Both CLI baselines resolved to gpt-4o-2024-11-20. For Claude Code, the API-

translation proxy mapped claude-sonnet-4-6 to GPT-4o through modelOverrides without rewriting prompts, adding tools, or expanding file access.

Evaluation setup. We compute the weighted score S_{total} across all test cases using the metrics in Section 3.6. GPT-4o is the primary evaluator for $S_{fact,lm}$, S_{logic} , and S_{ins} , creating potential evaluator dependence because it also participates in goal extraction and serves as the DataCrossAgent backbone. We mitigate, but do not eliminate, this risk through deterministic checks, a blind 50-task human audit, and cross-judge agreement.

Scoring consistency. The human audit yields Spearman correlations of 0.72, 0.69, and 0.61 for Completeness, Logic, and Insightfulness. Across GPT-4o, Gemini-2.5-Flash, and Qwen3-VL-32B-Instruct, averaged scores achieve $ICC(2,k) = 0.8159$ (95% CI [0.6174, 1.0000]; $F(199, 398) = 21.85$, $p < 0.001$), with small pairwise Bland-Altman biases (+0.0536, +0.0611, and +0.0075). This supports stable aggregate comparisons without implying full judge interchangeability; complete analyses are in Appendix H and Appendix D.

5.2 Main Results

Table 2 evaluates DataCrossAgent against various baselines. DataCrossAgent achieves competitive overall performance among strong agentic baselines while showing clear gains over vanilla LMMs, especially on completeness and hard-task robustness.

Benchmark Discriminative Power. DataCrossBench exhibits strong **discriminative power** through its stratified Easy/Hard task design. Most evaluated systems show an Easy-to-Hard performance decay (Δ), although the magnitude varies by model. For example, Qwen3-vl drops by 16.52 percentage points in Overall, Codex CLI drops by 10.68 percentage points, and GPT-4o shows a smaller 3.98 percentage-point drop. This pattern suggests that the Hard set introduces reasoning challenges beyond simple pattern matching, while avoiding the unsupported claim that every model degrades uniformly.

Competitive Agent Performance. DataCrossAgent remains competitive under a substantially stronger baseline set, reaching an Overall score of 0.5172. The comparison now includes two general-purpose CLI agent systems, Codex CLI and Claude CLI, which perform iterative planning, file inspection, code execution, and tool-assisted analysis over heterogeneous

Table 2: **Four-dim Insight Score.** Blue badges highlight per-column best scores and the average profile of DataCrossAgent for faster visual scanning. The Four-dim Insight Score is normalized to $[0, 1]$, where higher values represent better performance. “Easy” refers to tasks involving only structured or textual data, while “Hard” denotes challenges requiring the parsing and integration of visual document images (e.g., scanned tables and charts).

	Factuality			Completeness			Logic			Insightfulness			Overall		
	Easy	Hard	Avg.	Easy	Hard	Avg.	Easy	Hard	Avg.	Easy	Hard	Avg.	Easy	Hard	Avg.
<i>General LLMs (Closed-Source & Open-Source)</i>															
GPT-4o	0.3313	0.3269	0.3291	0.4689	0.4160	0.4425	0.4376	0.3815	0.4096	0.3835	0.3272	0.3554	0.4000	0.3602	0.3801
Gemini-2.5-flash	0.2957	0.3198	0.3078	0.3672	0.3517	0.3594	0.4063	0.3902	0.3982	0.3930	0.3892	0.3911	0.3600	0.3592	0.3596
Qwen3-vl	0.3074	0.1825	0.2473	0.4112	0.2328	0.3220	0.4337	0.2436	0.3398	0.4486	0.2680	0.3562	0.3939	0.2287	0.3117
<i>Specialized Agent Frameworks</i>															
AgentPoirot (GPT-4o)	0.3069	0.2391	0.2730	0.4409	0.3851	0.4130	0.5652	0.5683	0.5668	0.5517	0.5445	0.5481	0.4533	0.4178	0.4356
ReActInsight (GPT-4o)	0.3563	0.3212	0.3388	0.6011	0.5851	0.5932	0.4040	0.4211	0.4125	0.6976	0.6975	0.6975	0.5124	0.5012	0.5068
<i>General-Purpose CLI Agent Systems</i>															
Codex CLI (GPT-4o)	0.4776	0.3347	0.4062	0.5545	0.4515	0.5030	0.6880	0.6068	0.6474	0.6200	0.5322	0.5761	0.5745	0.4677	0.5211
Claude CLI (GPT-4o)	0.3933	0.4674	0.4303	0.4390	0.5400	0.4895	0.4698	0.5854	0.5276	0.4646	0.5725	0.5186	0.4378	0.5354	0.4866
<i>Proposed Framework</i>															
DataCrossAgent (GPT-4o)	0.4508	0.4031	0.4270	0.5720	0.5368	0.5544	0.5912	0.5645	0.5779	0.5462	0.5326	0.5394	0.5331	0.5012	0.5172

Table 3: Ablation Study Results on Four-dimension Insight Score. “w/o” indicates “without”; the Four-dim Insight Score ranges from 0 to 1, and higher values indicate better performance.

	Four-dim Insight Score				
	Factuality	Completeness	Logic	Insightfulness	Overall
Full DataCrossAgent	0.4268 ± 0.1493	0.5543 ± 0.0849	0.5778 ± 0.1085	0.5394 ± 0.0837	0.5170 ± 0.0655
<i>Ablation Experiments</i>					
w/o reReAct	0.3407 ± 0.1597	0.5017 ± 0.0862	0.4886 ± 0.0916	0.4652 ± 0.0858	0.4417 ± 0.0719
w/o Key-identification	0.3917 ± 0.1543	0.5385 ± 0.0835	0.5378 ± 0.1235	0.4956 ± 0.0936	0.4836 ± 0.0699
w/o Cross-pollination	0.3908 ± 0.1563	0.5320 ± 0.0904	0.5357 ± 0.1100	0.4840 ± 0.0785	0.4784 ± 0.0691
w/o Key & Cross	0.3283 ± 0.1576	0.4691 ± 0.0943	0.4959 ± 0.0831	0.4354 ± 0.0806	0.4238 ± 0.0721

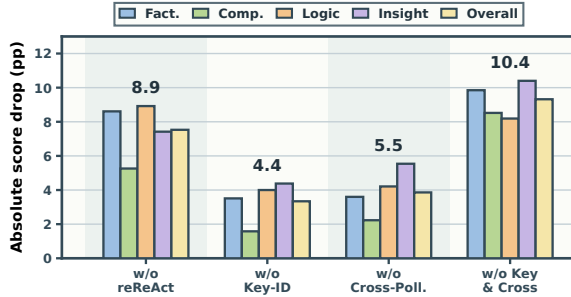


Figure 3: Ablation impact relative to the full DataCrossAgent. Bars report absolute score drops, in percentage points, across the five evaluation dimensions.

local inputs. While these off-the-shelf agentic baselines attain higher peaks in several aggregate columns, DataCrossAgent still delivers a clear **Factuality** gain over vanilla GPT-4o: its average factuality rises from 0.3291 to 0.4270, a relative improvement of **29.7%**. This indicates that the framework remains effective at suppressing numerical hallucinations while preserving a balanced profile across the four evaluation dimensions.

Robustness and Stability. Beyond peak scores,

DataCrossAgent demonstrates comparatively stable behavior across varying task complexities. Several baselines exhibit marked difficulty sensitivity, such as Qwen3-vl with a 16.52 percentage-point absolute drop in Overall and Codex CLI with a 10.68 percentage-point drop, whereas DataCrossAgent maintains a comparatively low decay rate ($\Delta = -0.0319$). This pattern suggests that the framework can retain a balanced profile as source heterogeneity increases, but it does not establish robustness to uncontrolled production noise.

Cost-Quality Profile. We ran a fresh 200-task companion profile with the same visible evidence, output schema, and scoring implementation across systems. DataCrossAgent reaches Overall 0.5193 at an estimated USD 0.1061 per task, compared with 0.5188 at USD 0.1630 for Claude Code and 0.5037 at USD 0.3184 for ReActInsight. Its mean runtime is 64.99 seconds per task, versus 75.00 and 141.20 seconds, respectively. These results position DataCrossAgent as a competitive-quality, lower-cost reference framework; because Codex CLI was not included in this companion profile, we

do not claim a global Pareto optimum.

Table 4: Compact cost–quality profile used in the companion analysis.

System	Overall	Runtime (s)	Cost/task
GPT-4o direct	0.4008	9.59	USD 0.0173
ReActInsight	0.5037	141.20	USD 0.3184
DataCrossAgent	0.5193	64.99	USD 0.1061
Claude Code	0.5188	75.00	USD 0.1630

External Transfer Check. On the independent 100-case UniDataBench suite, DataCrossAgent obtains average Insight 0.4816 and Summary 0.6366, versus 0.4750 and 0.5792 for the reported ReActInsight baseline. It is slightly lower on Medium Insight (0.4687 vs. 0.4704) but higher on Easy/Hard Insight and every Summary split. This is preliminary evidence of transfer beyond the DataCrossBench construction pipeline, not evidence of exhaustive enterprise-domain generalization.

Visual-Dependence Diagnostic. On the same 100 Hard tasks, removing all images lowers Claude CLI’s Overall from 0.5354 to 0.4119, an absolute drop of 0.1235 (23.07% relative); Factuality drops from 0.4674 to 0.3710 and Completeness from 0.5400 to 0.4280. The result supports the interpretation of Hard as image-dependent, while the full protocol and table are reported in Appendix F.

5.3 Ablation Experiments

To quantify the contribution of each module, we conducted an ablation study as shown in Table 3.

Effect of reReAct. Removing the **reReAct** mechanism (*w/o reReAct*) results in the most dramatic performance decline, with the Overall score dropping from 0.5170 to 0.4417. Most notably, *Logic* and *Factuality* show relative decreases of 15.4% and 20.1% respectively. This result indicates that reasoning stability depends on the dual-loop structure, which separates strategic decomposition from tactical code debugging. Without this separation, the system is prone to cascading errors where a single syntax failure or misinterpreted data row derails the entire analysis.

Effect of Key Source Identification and Cross-Pollination. The exclusion of Key-identification and Cross-pollination (*w/o Key & Cross*) reduces the agent’s performance to a level near vanilla GPT-4o (0.4238 vs 0.3801), reflecting two complementary failures: (i) **Key-identification** acts as a precision filter, and removing it fragments the agent’s

attention across less relevant sources, leading to lower **Completeness**; (ii) **Cross-pollination** contributes to **Insightfulness**, and removing it makes the agent more likely to generate localized single-source insights rather than non-trivial patterns that emerge from synthesizing disparate sources, such as visual dashboard trends and structured sales logs.

Cross-checks and error profile. The companion diagnostics provide a complementary view of where the framework succeeds and fails. Removing images from the Hard split lowers Claude CLI’s Overall from 0.5354 to 0.4119, confirming that the split measures visual evidence integration rather than image presence alone. On the independent UniDataBench suite, DataCrossAgent improves Summary from 0.5792 to 0.6366 over the reported ReActInsight baseline, while the smaller Insight gain (0.4750 to 0.4816) indicates that transfer is positive but not uniform across metrics. The cost profile further shows that DataCrossAgent’s quality is achieved without the highest runtime or estimated cost.

These results suggest three recurring failure modes: visual extraction and OCR errors when evidence is image-dependent, incomplete cross-source alignment when entities or units differ, and cascading tool failures when exploratory code is not verified. The ablations and judge-consistency checks indicate that reReAct, source prioritization, cross-pollination, and executable verification address different parts of this error profile; no single component eliminates all failure cases.

6 Conclusion

We present DataCross, a controlled and reproducible benchmark-and-agent framework for heterogeneous analytics over structured sources and visual zombie data. DataCrossBench provides verifiable cross-source tasks, while DataCrossAgent combines source profiling, Reasoning ReAct (reReAct), cross-source pollination, and executable verification. Experiments show that DataCrossAgent improves factuality over direct GPT-4o prompting, provides a competitive quality–cost profile in the profiled systems, and transfers preliminarily to an external benchmark. These results support cross-modal heterogeneous analysis as a practical research direction, while not constituting a complete evaluation of production enterprise analytics.

Limitations

DataCrossBench is a controlled and reproducible testbed for cross-source and cross-modal evidence integration, rather than a complete simulation of production analytics. Although its seeds are curated from industry reports, the released CSV, SQLite, JSON, TXT, and image artifacts are generated by executable Python and SQL programs and filtered through automated and human verification. This improves auditability and executability, but does not systematically model OCR corruption, missing documents, schema drift, open-ended contradictory records, or multilingual reports.

The benchmark contains 200 end-to-end tasks and prioritizes task complexity and verification quality over raw scale. Its domain distribution is uneven, including 15 AI Healthcare tasks and 23 Short-form Drama tasks. These subsets are representative vertical probes, not sufficient evidence for standalone domain-level generalization claims. Future releases should expand these under-represented domains and include additional report genres and corruption settings.

Several evaluation limitations remain. GPT-4o is used in goal extraction, as the DataCrossAgent backbone, and as one component of semantic scoring. Human alignment audits and cross-model agreement reduce, but do not eliminate, evaluator dependence. DataCrossAgent is also a task-specific composition of established agentic mechanisms and does not dominate every baseline on every metric. Its results should therefore be interpreted as evidence of competitive quality and cost efficiency in this setting, rather than as evidence of a universally superior reasoning method.

The current benchmark is file-based and does not evaluate authentication, access control, live database or API connections, query latency, query cost, or data governance. DataCrossAgent maintains task-local state only; it has no persistent memory across tasks and does not learn from previous failures. Future work should extend the benchmark with live-source integration, permission failures, large-schema discovery, latency and cost budgets, governance requirements, persistent-memory protocols, abstention behavior, and explicit handling of corrupted or contradictory evidence.

Ethical Considerations

The released benchmark artifacts are synthesized and contain no personally identifying informa-

tion or offensive content. Human raters evaluated anonymized model outputs, and the benchmark is intended for research evaluation rather than unrestricted deployment.

Acknowledgments

This work is supported by National Natural Science Foundation of China (92470121, 62402016), Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China (JYB2025XDXM113), National Key R&D Program of China (2024YFA1014003), Zhongguancun Academy (C20250204, C20250602), Beijing Major Science and Technology Project (Z251100008125043, Z251100008425023), and High-performance Computing Platform of Peking University.

References

- Anthropic. 2026. Claude Code CLI Reference. <https://code.claude.com/docs/en/cli-usage>. Accessed: 2026-05-26.
- Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhi-fang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, and 45 others. 2025. *Qwen3-vl technical report*. Preprint, arXiv:2511.21631.
- Yew Ken Chia, Liying Cheng, Hou Pong Chan, Maojia Song, Chaoqun Liu, Mahani Aljunied, Soujanya Poria, and Lidong Bing. 2025. *M-longdoc: A benchmark for multimodal super-long document understanding and a retrieval-aware tuning framework*. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 9233–9250, Suzhou, China. Association for Computational Linguistics.
- Alex Egg, Martin Iglesias Goyanes, Friso Kingma, Andreu Mora, Leandro von Werra, and Thomas Wolf. 2025. Dabstep: Data agent benchmark for multi-step reasoning. *arXiv preprint arXiv:2506.23719*.
- Negar Foroutan, Angelika Romanou, Matin Ansaripour, Julian Martin Eisenschlos, Karl Aberer, and Rémi Lebret. 2025. *WikiMixQA: A multimodal benchmark for question answering over tables and charts*. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 24941–24958, Vienna, Austria. Association for Computational Linguistics.
- Google. 2025. *Gemini 2.5 Flash*. Accessed: 2026-05-25.
- Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang,

- Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. [Deepseek-r1 incentivizes reasoning in llms through reinforcement learning](#). *Nature*, 645(8081):633–638.
- Siyuan Guo, Cheng Deng, Ying Wen, Hechang Chen, Yi Chang, and Jun Wang. 2024. [Ds-agent: Automated data science by empowering large language models with case-based reasoning](#). In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 16813–16848. PMLR.
- Xinyi He, Mengyu Zhou, Xinrun Xu, Xiaojun Ma, Rui Ding, Lun Du, Yan Gao, Ran Jia, Xu Chen, Shi Han, Zejian Yuan, and Dongmei Zhang. 2024. [Text2analysis: a benchmark of table question answering with advanced data analysis and unclear queries](#). In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI’24/IAAI’24/EAAI’24. AAAI Press.
- Xueyu Hu, Ziyu Zhao, Shuang Wei, Ziwei Chai, Qianli Ma, Guoyin Wang, Xuwu Wang, Jing Su, Jingjing Xu, Ming Zhu, Yao Cheng, Jianbo Yuan, Jiwei Li, Kun Kuang, Yang Yang, Hongxia Yang, and Fei Wu. 2024. [InfiAgent-DABench: Evaluating agents on data analysis tasks](#). In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 19544–19572. PMLR.
- Martin Just, Petra Schubert, Jonas Blatt, and Patrick Delfmann. 2024. [Data preprocessing for cross-system analysis: The daproxsa approach](#). *Procedia Computer Science*, 239:1635–1644. CENTERIS – International Conference on ENTERprise Information Systems / ProjMAN - International Conference on Project MANAGEMENT / HCist - International Conference on Health and Social Care Information Systems and Technologies 2023.
- Somnath Kumar, Yash Gadhia, Tanuja Ganu, and Akshay Nambi. 2024. [Mmctagent: Multi-modal critical thinking agent framework for complex visual reasoning](#). *arXiv preprint arXiv:2405.18358*.
- Yuhang Lai, Chengxi Li, Yiming Wang, Tianyi Zhang, Ruiqi Zhong, Luke Zettlemoyer, Wen-Tau Yih, Daniel Fried, Sida Wang, and Tao Yu. 2023. [DS-1000: A natural and reliable benchmark for data science code generation](#). In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 18319–18345. PMLR.
- Hongxin Li, Jingran Su, Yuntao Chen, Qing Li, and ZHAO-XIANG ZHANG. 2023. [Sheetcopilot: Bringing software productivity to the next level through large language models](#). In *Advances in Neural Information Processing Systems*, volume 36, pages 4952–4984. Curran Associates, Inc.
- Jinwei Lu, Yuanfeng Song, Chen Zhang, and Raymond Chi-Wing Wong. 2026. [Multivis-agent: A multi-agent framework with logic rules for reliable and comprehensive cross-modal data visualization](#). *Proc. ACM Manag. Data*, 4(1).
- Bodhisattwa Prasad Majumder, Harshit Surana, Dhruv Agarwal, Bhavana Dalvi Mishra, Abhijeetsingh Meena, Aryan Prakhar, Tirth Vora, Tushar Khot, Ashish Sabharwal, and Peter Clark. 2025. [Discoverybench: Towards data-driven discovery with large language models](#). In *International Conference on Learning Representations (ICLR)*.
- Abhijit Manatkar, Devarsh Patel, Hima Patel, and Naresh Manwani. 2025. [Ilaeda: An imitation learning based approach for automatic exploratory data analysis](#). In *Proceedings of the 4th International Conference on AI-ML Systems*, AIMLSys’24, New York, NY, USA. Association for Computing Machinery.
- Van-Quang Nguyen and Takayuki Okatani. 2026. [CoReTab: Improving multimodal table understanding with code-driven reasoning](#). In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6498–6523, Rabat, Morocco. Association for Computational Linguistics.
- OpenAI. 2024. [Gpt-4o system card](#). System card, OpenAI. ArXiv:2410.21276.
- OpenAI. 2025. [Introducing GPT-5](#). <https://openai.com/index/introducing-gpt-5/>. Accessed: 2026-05-25.
- OpenAI. 2026. [Codex CLI](#). <https://developers.openai.com/codex/cli>. Accessed: 2026-05-26.
- Xiangyu Peng, Can Qin, Zeyuan Chen, Ran Xu, Caiming Xiong, and Chien-Sheng Wu. 2025. [UNIDOC-BENCH: A unified benchmark for document-centric multimodal RAG](#). *arXiv preprint arXiv:2510.03663*.
- Gaurav Sahu, Abhay Puri, Juan A. Rodriguez, Amirhossein Abaskohi, Mohammad Chegini, Alexandre Drouin, Perouz Taslakian, Valentina Zantedeschi, Alexandre Lacoste, David Vazquez, Nicolas Chapados, Christopher Pal, Sai Rajeswar, and Issam Laradji. 2025. [Insightbench: Evaluating business analytics agents through multi-step insight generation](#). In *International Conference on Learning Representations*, volume 2025, pages 4683–4715.
- Xingyu Shen, Shengding Hu, Xinrong Zhang, Xu Han, Xiaojun Meng, Jiansheng Wei, Zhiyuan Liu, and Maosong Sun. 2025. [Autoclean: Llms can prepare their training corpus](#). In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (System Demonstrations)*, pages 85–95, Albuquerque, New Mexico. Association for Computational Linguistics.

- Anshul Singh, Chris Biemann, and Jan Strich. 2025. [Mtabvqa: Evaluating multi-tabular reasoning of language models in visual space](#). In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 19866–19891, Suzhou, China. Association for Computational Linguistics.
- Sanchit Sinha, Oana Frunza, Kashif Rasul, Yuriy Nevmyvaka, and Aidong Zhang. 2025. [Chart-rvr: Reinforcement learning with verifiable rewards for explainable chart reasoning](#). *arXiv preprint arXiv:2510.10973*.
- Li Sun, Liu He, Shuyue Jia, Yangfan He, and Chenyu You. 2025. [DocAgent: An agentic framework for multi-modal long-context document understanding](#). In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 17701–17716, Suzhou, China. Association for Computational Linguistics.
- Navve Wasserman, Roi Pony, Oshri Naparstek, Adi Raz Goldfarb, Eli Schwartz, Udi Barzelay, and Leonid Karlinsky. 2025. [REAL-MM-RAG: A real-world multi-modal retrieval benchmark](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 31660–31683, Vienna, Austria. Association for Computational Linguistics.
- Han Weng, Zhou Liu, Yuanfeng Song, Xiaoming Yin, Xing Chen, and Wentao Zhang. 2026. [UniDataBench: Evaluating data analytics agents across structured and unstructured data](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 33755–33780, San Diego, California, United States. Association for Computational Linguistics.
- An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. [Qwen3 technical report](#). *arXiv preprint arXiv:2505.09388*.
- Wenqi Zhang, Yongliang Shen, Zeqi Tan, Guiyang Hou, Weiming Lu, and Yueting Zhuang. 2023. [Data-Copilot: Bridging billions of data and humans with autonomous workflow](#). *arXiv preprint arXiv:2306.07209*.
- Yuge Zhang, Qiyang Jiang, Xingyu Han, Nan Chen, Yuqing Yang, and Kan Ren. 2024. [Benchmarking data science agents](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5677–5700, Bangkok, Thailand. Association for Computational Linguistics.
- Zhenghao Zhu, Chuxue Cao, Sirui Han, Yuanfeng Song, Xing Chen, Caleb Chen Cao, and Yike Guo. 2025. [Medinsightbench: Evaluating medical analytics agents through multi-step insight discovery in multimodal medical data](#). *arXiv preprint arXiv:2512.13297*.

Appendix

Appendix Roadmap

A	Problem Formulation
B	Human-in-the-Loop Construction and Verification
C	Illustrative Output Example
D	Case Study and Human Alignment Audit
E	Additional Benchmark Documentation
F	Baseline Details
G	Details of Hybrid Priority Scoring
H	Scoring Consistency Analysis
I	Multi-Agent System Prompt Templates

A Problem Formulation

Setting. We study data analysis in organizations where relevant information is distributed across *heterogeneous* sources: some are directly queryable (e.g., SQL tables), while others are only available as *visual documents* (e.g., scanned reports or screenshots containing tables). Existing analytics agents often perform well when the answer resides in a single structured source, but struggle to *jointly reason* over structured data and table content embedded in images.

Data sources. Let $\mathcal{M} = \{\text{SQL}, \text{CSV}, \text{JSON}, \text{TXT}, \text{IMG}\}$ denote the modality set, and let $\mathcal{S} = \{s_i\}_{i=1}^n$ denote the concrete source instances available to a task. Each source has a modality $m(s_i) \in \mathcal{M}$: SQL sources are relational databases, CSV sources are delimited files, JSON sources are semi-structured stores, TXT sources are unstructured text, and IMG sources are images or scanned PDFs containing *tables* and other visually formatted records. We emphasize that sources with $m(s_i) = \text{IMG}$ cannot be reliably queried without extracting table structure (rows/columns/headers) and cell values from pixels.

Input. The input to the system is:

$$X = \langle \mathcal{S}, G \rangle, \quad (4)$$

where G is a natural-language analysis goal (a question or a higher-level objective), e.g., “Summarize the key cost drivers in 2023 and explain discrepancies between the finance database and the scanned invoices.”

Output. The system produces a response:

$$Y = \langle \text{Summary}, \text{Insights}, \text{Evidence} \rangle. \quad (5)$$

Summary is a concise natural-language overview of the findings. Insights is a list of specific, actionable findings (e.g., anomalies, trends, mismatches,

root-cause hypotheses). Evidence provides traceable support for each insight by pointing to the originating sources, such as SQL rows/fields, CSV row indices, JSON paths, text spans, or image regions / extracted table cells.

Task objective (Unified Analysis). Given $X = \langle \mathcal{S}, G \rangle$, the goal is to generate Y such that:

- **Cross-source retrieval:** the system can access and use information from any subset of $\mathcal{S}$ required by G ;
- **Cross-modal alignment:** data and column names mentioned or shown in sources with $m(s_i) = \text{IMG}$ can be matched to records in structured sources (e.g., SQL/CSV/JSON) when they refer to the same real-world item;
- **Joint reasoning for insights:** insights may require combining evidence from both structured data and visual tables;
- **Evidence grounding:** each reported insight is supported by explicit evidence links in Evidence, enabling verification.

Why this is non-trivial. The main difficulty is that IMG sources are not directly queryable: answering G may require (i) perception and table extraction from images, (ii) normalization of extracted values (units, formats, OCR noise), and (iii) reliable alignment/join operations between extracted visual records and structured tables. Our benchmark and agent are designed to evaluate and enable this end-to-end capability.

B Human-in-the-Loop Construction and Verification

To ensure DATACROSSBENCH captures key properties of real-world industrial scenarios while maintaining strict ground-truth accuracy, we implemented a two-stage *Human-in-the-Loop (HITL)* pipeline. This process separates data construction (via AI-assisted synthesis) from quality assurance (via double-blind human verification).

B.1 Stage 1: AI-Assisted Reverse-Synthesis

Constructing heterogeneous datasets where structured files (SQL/CSV) perfectly correlate with unstructured visual documents (PDF/Images) is challenging. We developed a **Data Generator Platform** (Figure 4) to streamline this via a “Reverse-Synthesis” workflow:

- **Insight-Driven Generation (Fig. 4a):** The workflow begins with a seed financial report. An LLM (GPT-4o) first extracts high-level insights and industry trends. Human experts then inject specific constraints (e.g., “Generate a non-trivial sales fact table and auxiliary dimensions that contradict the PDF summary by 5%”).
- **Code-Based Artifact Creation (Fig. 4b):** Instead of asking the LLM to output raw data directly (which causes hallucinations), the platform generates executable Python/SQL scripts. These scripts synthesize the raw files and render the corresponding visual charts/tables, ensuring mathematical consistency between the “zombie data” (images) and structured sources.

B.2 Stage 2: Double-Blind Quality Verification

All synthesized tasks undergo a rigorous review using the **DataCrossBench Verification Platform** (Figure 5). This stage ensures that the generated “zombie data” is legible and that the cross-source logic is sound.

- **Unified Heterogeneous View (Fig. 5a):** The interface provides a “God’s eye view” of the task. The left panel displays the ground-truth metadata and reasoning chains. The center and right panels allow annotators to inspect structured data (SQL/CSV previews) alongside the generated visual documents (charts/scanned tables) to verify cross-modal alignment.
- **Granular Scoring System (Fig. 5b):** Annotators evaluate each task on a 0-10 scale across three dimensions: *Overall Score*, *Data Quality* (schema correctness), and *Image Quality* (OCR legibility). The “Save Review” module enforces a feedback loop; tasks scoring below 8.0 are flagged with specific comments (e.g., “Image resolution too low for OCR”) and returned to the synthesis stage for regeneration.

C Illustrative Output Example

To make the expected deliverable concrete, Listing 1 shows a shortened illustrative output excerpt that follows the schema used in our benchmark evaluation. The example is anonymized and truncated

for space, but preserves the structure of summary, grounded insights, and cross-dataset discoveries.

Listing 1: Illustrative generated report excerpt

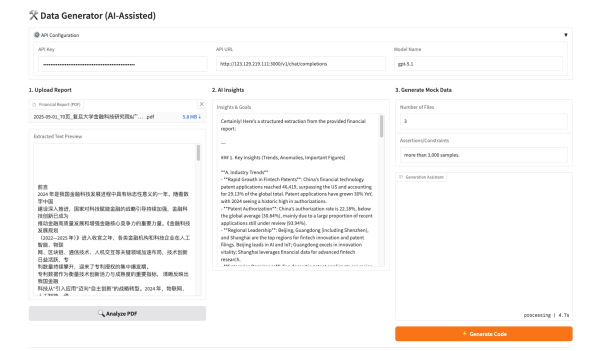
```
{
  "summary": "A discrepancy emerges between the SQL expense
  ↳ ledger and a scanned
  ↳ invoice table for Q3 warehousing costs. The structured
  ↳ records show a stable
  ↳ monthly baseline, but the visual invoice evidence reveals a
  ↳ repeated surcharge
  ↳ applied to the same vendor cluster.",
  "insights": [
    "Vendor group A accounts for the majority of the
    ↳ discrepancy after unit
    ↳ normalization.",
    "The anomaly appears only after aligning invoice dates
    ↳ with the fiscal
    ↳ calendar used in the SQL ledger."
  ],
  "cross_dataset_discoveries": [
    "The dashboard-level trend spike is not explained by
    ↳ higher sales volume;
    ↳ it is driven by a reporting mismatch between two source
    ↳ systems."
  ],
  "limitations": [
    "One image-based table contains OCR ambiguity in a low-
    ↳ resolution cell."
  ],
  "next_steps": [
    "Reconcile the vendor mapping table and re-run the
    ↳ executable verification
    ↳ script on the flagged records."
  ]
}
```

D Execution Trajectory and Human Alignment Audit

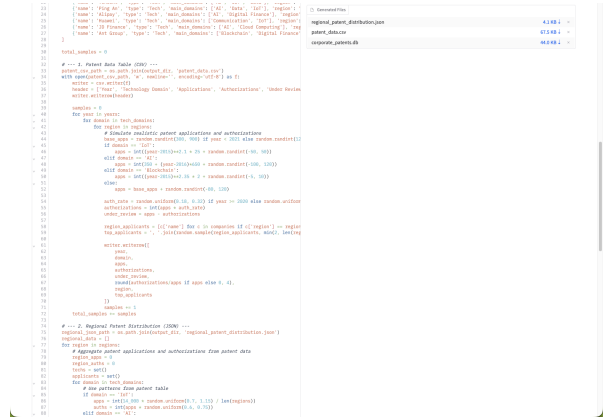
This appendix presents a representative successful execution trajectory to illustrate the cross-source grounding process in detail. Furthermore, it details the human-LLM alignment audit conducted to validate the reliability of our judge-dependent scoring dimensions.

D.1 Representative Trajectory: Grounded Cross-Source Market Analysis

This case demonstrates an effective cross-source grounding trajectory: the agent profiles heterogeneous data sources, elevates source-level observations into executable cross-source checks, and synthesizes a final market conclusion directly backed by retained artifacts. The complete execution chain is mapped in Figure 6. Notably, the generated order-amount visualization used in the platform-comparison branch (Figure 6c), alongside the preserved analysis script and stat.json file, provides a transparent audit trail. This transparency allows users to mathematically verify intermediate claims—such as average order amounts and retention rates—before accepting the broader synthesized market narrative.

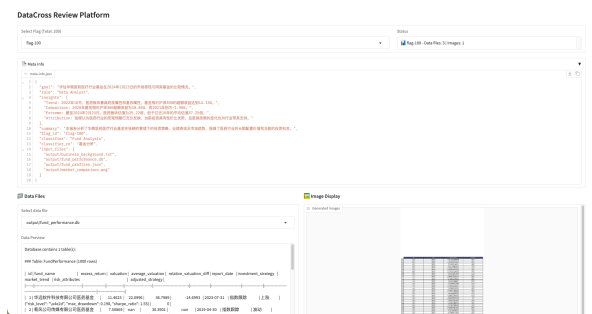


(a) Insight-Driven Generation

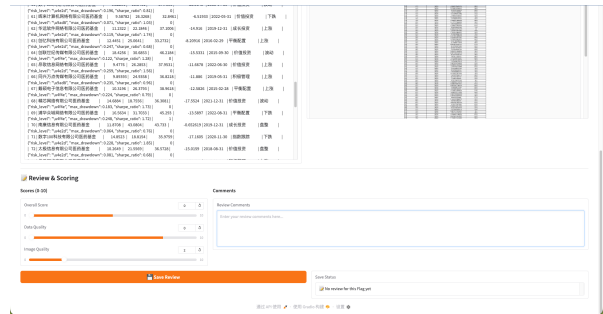


(b) Code-Based Artifact Creation

Figure 4: **The AI-Assisted Data Generator Platform.** This interface facilitates the “Reverse-Synthesis” pipeline, allowing experts to produce large-scale, heterogeneous datasets that adhere to specific logical constraints.



(a) Unified Heterogeneous View



(b) Granular Scoring System

Figure 5: **The DataCrossBench Verification Platform.** This tool is used for the final quality audit, ensuring that the synthesized heterogeneous data supports the intended analysis goals without logical flaws or artifacts.

D.2 Human and LLM Alignment Audit

To directly test whether the LLM-judge scores align with human assessment, we randomly sampled 50 tasks from DataCrossBench while preserving the Easy/Hard balance. For each sampled task, three human raters independently evaluated anonymized model outputs under a blind protocol: model identities, LLM-judge scores, and S_{total} rankings were hidden from the raters. We then computed rank correlations between the average human score and the corresponding LLM-judge score for each dimension, together with inter-rater reliability and preference agreement with the S_{total} ranking.

E Additional Benchmark Documentation

E.1 Source Report Transparency

This subsection explains how seed reports were selected before synthetic artifacts were generated. The goal is to make the benchmark origin auditable without exposing proprietary report contents. Table 6 summarizes the report archetypes, the visual

evidence they typically contain, and the construction role each archetype plays.

E.2 Task Characterization Statistics

This subsection provides the basic scale and composition statistics behind the benchmark. Table 7 gives a compact summary of the task count, modality coverage, and Easy/Hard split. Table 8 then reports the annotation-derived statistics used to verify that tasks require multi-source and cross-modal evidence rather than single-file lookup.

E.3 Representative Task Patterns

The final documentation block complements the aggregate statistics with concrete task-level evidence. Figure 7 visualizes the construction audit, while Figure 6 shows a representative task layout with structured files, textual context, visual evidence, and the expected reasoning path.

Task and Source Grounding

Case. Flag-140 asks: analyze the drivers of Meituan’s 18% revenue growth in 2024.

Loaded evidence. The run creates four usable agents from `dimension_data.db` and `fact_orders.csv`: *PerformanceMetrics*, *AttributionFactors*, *MetricFactorMapping*, and *fact_orders*.

(a) Task context and loaded multi-source evidence

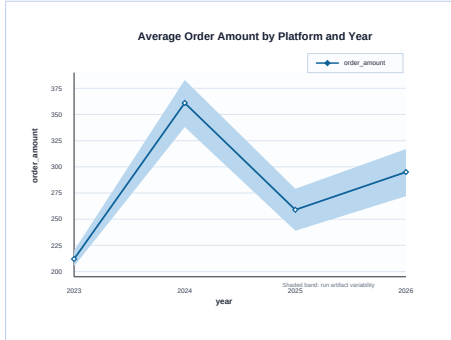
Executed Cross-Source Checks

City-tier demand: low-tier cities combine salient GTV-growth evidence with 271.37 RMB average order amount.

Platform comparison: in middle-tier cities, Meituan exceeds Douyin in order amount (354.88 vs. 247.36 RMB) and retention (8% vs. 3%).

Attribution: young users’ immediacy demand corresponds to 8% growth in low-tier cities, while middle-tier cities show higher order value (394.62 RMB).

(b) Retained metrics from executable crossover checks



(c) Generated visual artifact supporting numerical claims

Task-Specific Evidence Chain

1. Profile / rank sources.

The execution log initializes by loading the task goal, a short background.txt, three SQLite tables, and the 60-row `fact_orders.csv`; the malformed `user_profiles.json` is dropped without blocking the run. Hybrid profiling then evaluates objective richness and LLM relevance. Formal labeling prioritizes *PerformanceMetrics* and *fact_orders* as **Primary**, with *AttributionFactors* and *MetricFactorMapping* as **Secondary**. This focuses the crossover analysis on KPI growth and order facts, while secondary tables provide causal links.

2. Explore / source-level insight.json.

- *PerformanceMetrics*: high-tier cities show the highest avg growth (122.50), followed by low-tier (107.85) and middle-tier (59.78); low-tier GTV growth is the strongest single metric.
- *fact_orders*: avg order amount peaks in 2024 (353.38 RMB); Meituan (382.62 RMB) measurably exceeds Douyin (338.76 RMB). Middle-tier orders are highest on average and most dispersed (std 143.82).
- *AttributionFactors*: retains three unique operational descriptors: core high-value, young immediacy-demand, and local-life deepening.
- *MetricFactorMapping*: eight discrete metric-factor links establish a semantic bridge from numeric KPI rows to business drivers.

3. Cross / executable questions and findings.

- *Q1 city-tier GTV vs. order*: the generated script validates columns, filters GTV growth, aggregates orders by city tier, and joins the target tiers. The retained statistic successfully links low-tier GTV growth (122.50) with 271.37 RMB average order amount.
- *Q2 platform advantage*: the code isolates middle-tier orders and groups by platform, exporting `stat.json` and a plot. The artifact records Meituan at 354.88 RMB (8 retained orders) vs. Douyin at 247.36 RMB (3 retained orders), making the platform comparison strictly inspectable.
- *Q3 young-immediacy*: the code maps *MetricFactorMapping* to *AttributionFactors*, filters the “young immediacy” factor, merges back to KPI records, and profiles young-user orders: low-tier 8 orders / 272.75 RMB against middle-tier 4 orders / 394.62 RMB.

4. Synthesize / auditable conclusion.

The final log records 16 single-source insights and 3 crossover findings before writing `flag-140_result.json` with `status=success`. Consequently, the synthesis goes beyond restating the prompt: it explicitly traces high-line resilience, low-tier growth potential, Meituan’s middle-tier advantage, and the young-immediacy narrative back to the preserved `code.py`, `stat.json`, `insight.json`, and plot artifacts.

(d) Auditable cross-source execution chain

Grounded takeaway. Cross-source grounding transforms a generic market narrative into an auditable evidence chain: sources are systematically profiled, source-level observations are preserved, executable joins yield reproducible numeric evidence, and the synthesis states a market implication with visible support. The grounded conclusion reveals that high-line cities remain resilient, low-tier cities demonstrate significant growth potential, and Meituan holds a measurable middle-tier advantage over Douyin.

Figure 6: Representative trajectory for a grounded cross-source market analysis. Evidence is drawn from task flag-140 evaluated by gpt-4o: the agent instantiated four source-specific data profiles and completed three executable cross-source verification steps to produce an auditable conclusion.

F Baseline Details

This appendix clarifies the agent baselines used in Table 2. The goal is not to reproduce each system’s original benchmark protocol, but to define how each baseline is interpreted under the same DataCrossBench task interface.

AgentPoirot. AgentPoirot is treated as a specialized insight-mining agent baseline following InsightBench-style analytical report generation (Sahu et al., 2025). For DataCrossBench, heterogeneous inputs are organized into a form that the agent can inspect, and the agent is prompted with the same task objective used by other systems. This baseline represents an insight-oriented agent de-

signed for iterative analytical discovery, but it does not use DataCrossAgent’s source-prioritization or cross-source pollination modules.

ReActInsight. ReActInsight is treated as another specialized agent baseline for iterative question-driven insight generation (Weng et al., 2026). It receives the benchmark task objective and the available task inputs after lightweight organization into accessible files. This setting tests whether a generic ReAct-style insight workflow can recover cross-source findings without DataCrossAgent’s explicit source profiling, priority routing, or final executable verification loop.

Codex CLI. Codex CLI is used as a general-purpose command-line agent baseline. Official

Table 5: Human and LLM alignment audit over 50 Easy/Hard stratified tasks. We report rank correlation between human average scores and LLM-judge scores for each scoring dimension, plus inter-rater reliability and preference agreement with the S_{total} ranking.

Dimension	Slice	n	Raters	Spearman	Kendall	Reliability	Agreement/notes
Completeness	Easy/Hard strat.	50	3	0.72	0.55	ICC=0.78	Aligns with LLM judge
Logic	Easy/Hard strat.	50	3	0.69	0.51	ICC=0.75	Higher on executable tasks
Insightfulness	Easy/Hard strat.	50	3	0.61	0.44	ICC=0.68	More subjective dimension
Overall preference	Easy/Hard strat.	50	3	–	–	Kripp. $\alpha = 0.71$	76% agrees with S_{total} ranking

Table 6: Source-report archetypes and selection criteria used in DataCrossBench construction. The table summarizes the document patterns used to anchor task design; it does not expose proprietary reports.

Domain	Seed report archetype	Typical visual evidence	Why retained for benchmark construction
Finance	Periodic statement-style PDFs and audit-like summaries	Scanned tables, invoice-style records, budget charts	Supports discrepancy analysis, temporal aggregation, and cross-source reconciliation between ledgers and visual records
AI Healthcare	Healthcare AI adoption briefs and service-usage summaries	Utilization charts, provider comparisons, summary tables	Encourages alignment between structured metrics, provider metadata, and visually embedded operational evidence
Internet / Technology	Platform/company research notes and internet-development summaries	User segmentation charts, KPI dashboards, growth trend plots	Provides mixed chart/structured evidence for platform-level trend analysis and attribution
Content / Media	Micro-drama industry reports and market analyses	Audience distribution charts, policy/innovation comparisons, market-size plots	Supports cross-source reasoning over users, content performance, and visually summarized market signals

Table 7: High-level task characterization of DataCrossBench.

Statistic	Value
Tasks	200 end-to-end analysis tasks
Fine-grained categories	6 topical categories
Macro sectors	Finance 82; internet / technology 80; content / media 23; AI healthcare 15
Supported modalities	CSV, SQLite DB, JSON, TXT, Image
Declared input files	963 total files
Structured sources	573 total (171 CSV / 200 SQLite / 202 JSON)
Visual documents	194 images across 100 Hard tasks
Text briefs	196 TXT files
Average files per task	4.82 (median 4; p90 8)
Average modalities per task	4.09; 30 tasks use all 5 modalities
Average structured sources per task	2.87
Easy / Hard split	100 / 100 by image requirement

documentation describes Codex CLI as a local terminal coding agent that can read files, modify code, and run commands within the selected working directory (OpenAI, 2026). In our benchmark, it represents a strong off-the-shelf agentic workflow for local file inspection, planning, code execution,

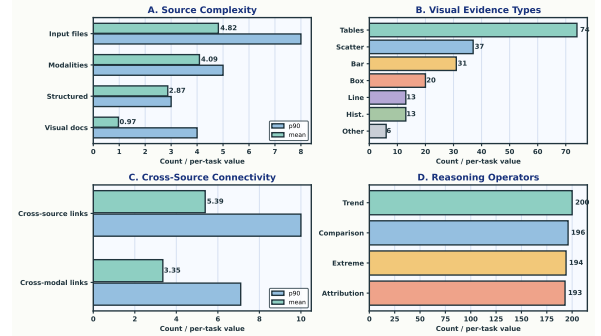


Figure 7: Benchmark characterization audit. The figure summarizes task-level construction statistics used to audit DataCrossBench, including source composition, modality coverage, cross-source links, visual evidence types, and reasoning targets. These statistics complement the aggregate benchmark table by showing how heterogeneous evidence and analytical operators are distributed across released tasks.

and report synthesis over heterogeneous task files. It is not customized with DataCrossAgent-specific modules.

Claude CLI. Claude CLI refers to the Claude Code command-line interface. Its official CLI documentation describes interactive and non-interactive sessions, file processing, tool configuration, and command-line execution modes (Anthropic, 2026).

Table 8: Detailed DataCrossBench characterization from benchmark construction annotations. Cross-source links count source pairs connected by annotated schema/entity anchors; cross-modal links are the subset spanning different modalities. Evidence-chain length counts evidence-bearing input sources per task.

Statistic	Total	Mean	Median	p90	Interpretation
Input files	963	4.82	4	8	Evidence-bearing sources; used as evidence-chain length
Modalities per task	–	4.09	4	5	Breadth over CSV, SQLite, JSON, TXT, and image inputs
Structured sources	573	2.87	3	3	CSV, SQLite, and JSON files used as queryable evidence
Visual documents	194	0.97	0.5	4	Image-based evidence concentrated in the 100 Hard tasks
Visual tables	74	0.37	0	1	Table-like images requiring visual extraction before analysis
Visual charts	120	0.60	0	4	Chart-like images requiring perceptual trend or comparison reading
Cross-source link pairs	1,078	5.39	3	10	Annotated schema/entity anchors connecting distinct input sources
Cross-modal link pairs	669	3.35	3	7.1	Cross-source anchors spanning different input modalities
Target insights	783	3.92	4	4	Ground-truth analytical findings attached to the task goals
Reasoning operators	783	3.92	4	4	Trend, comparison, extreme-value, and attribution reasoning targets

We include it as a second general-purpose CLI agent baseline to compare DataCrossAgent against a different off-the-shelf agentic system with local inspection and tool-use capabilities. As with Codex CLI, it is evaluated as a generic CLI agent rather than a benchmark-specific method.

F.1 Efficiency, Transfer, and Visual-Dependence Diagnostics

This subsection records the companion analyses used to clarify cost efficiency, visual dependence, and transfer beyond the DataCrossBench construction pipeline. All systems in the cost profile used the same 200 tasks, visible evidence, five-field output schema, and scoring implementation.

The frozen run configuration, per-case logs, version records, file whitelist, and baseline prompt bundle will be released alongside the benchmark artifact so that the shared protocol can be inspected end-to-end.

G Details of Hybrid Priority Scoring

This appendix provides the granular mathematical definitions for the components of the Hybrid Priority Scoring mechanism described in Section 4.

Objective Richness (S_{obj}). This score measures the intrinsic information density of the dataset. Let D be a dataset with rows R and columns C . The objective score is defined as:

$$S_{obj} = w_q \cdot S_{quality} + w_r \cdot S_{richness} + w_t \cdot S_{temp} \quad (6)$$

where $S_{quality} = 10 \cdot (1 - \text{missing_rate})$, and $S_{richness}$ is a composite of column count $|C|$, logarithmic row scaling $\log_{10}(|R|)$, and the average

unique value ratio $\bar{\rho}_{unique}$ across all dimensions. S_{temp} is a binary indicator (0 or 10) based on the presence of temporal keywords, prioritizing datasets that allow for longitudinal analysis.

Semantic Relevance (S_{sem}). To ensure the agent focuses on data pertinent to the user’s goal, we calculate the overlap between the goal’s keyword set K_{goal} and the data schema’s keyword set K_{schema} , where λ is a scaling factor:

$$S_{sem} = \min \left(\lambda \cdot \frac{|K_{goal} \cap K_{schema}|}{|K_{goal}|}, 10 \right) \quad (7)$$

Subjective Utility (S_{LLM}). S_{LLM} is derived from the Coordinator Agent’s high-level reasoning. The agent evaluates the data source description against the global task context, providing a score that captures implicit utility that structural metrics might overlook.

Total Priority Score. The final importance rank $S_{priority}$ is a weighted aggregation:

$$S_{priority} = 0.4 \cdot S_{obj} + 0.3 \cdot S_{sem} + 0.3 \cdot S_{LLM} \quad (8)$$

During execution, the coordinator estimates S_{obj} , S_{sem} , and S_{LLM} for each source, computes $S_{priority}$, and ranks sources by descending $S_{priority}$ before cross-pollination.

H Scoring Consistency Analysis

To validate the reliability and interchangeability of the scoring results across multiple Large Language Models (GPT-4o, Gemini-2.5-Flash, and Qwen3-VL-32B-Instruct), we conducted a quantitative consistency analysis using the **Intraclass Correlation Coefficient (ICC)** and **Bland-Altman plot analysis**.

Table 9: Cost-efficiency companion profile over the full 200-task benchmark. Estimated cost uses fixed GPT-4o rates of USD 2.50 per million input tokens and USD 10.00 per million output tokens.

System	Task set	Overall	Runtime/task (s)	API calls/task	Input tokens/task	Output tokens/task	Est. cost/task	Parse success
GPT-4o direct prompting	Full (200)	0.4008	9.59	1.00	5,810.98	276.64	USD 0.0173	100.0%
ReActInsight	Full (200)	0.5037	141.20	28.32	88,446.14	9,731.51	USD 0.3184	100.0%
DataCrossAgent	Full (200)	0.5193	64.99	15.52	24,733.84	4,430.05	USD 0.1061	100.0%
Claude Code 2.1.107	Full (200)	0.5188	75.00	7.36	61,969.25	806.24	USD 0.1630	98.0%

Table 10: No-image diagnostic on the same 100 Hard tasks.

Setting	Visible inputs	Tasks	Factuality	Completeness	Overall
Full-input Hard	SQL/CSV/JSON/TXT + images	100	0.4674	0.5400	0.5354
No-image Hard	SQL/CSV/JSON/TXT only	100	0.3710	0.4280	0.4119
Overall drop: 0.1235 absolute (23.07% relative).					

H.1 Inter-Rater Reliability: ICC(2,k)

We employed the **ICC(2,k)** model (two-way random effects, absolute agreement, averaged measures) to assess the reliability of scores averaged across $k = 3$ judge models over $n = 200$ tasks. This model treats both the subjects and the raters as random effects, accounting for both systematic biases and random errors.

The statistical analysis yielded an **ICC(2,k)** estimate of **0.8159**, with a 95% confidence interval ranging from **0.6174 to 1.0000**. The associated F-test confirmed the significance of these results ($F(199, 398) = 21.8505, p < 0.001$). Under the conventional interpretation of ICC, this range indicates **good reliability**, suggesting that the judge-dependent dimensions used in our benchmark are stable across model families.

H.2 Bland–Altman Consistency Analysis

To further investigate potential systematic bias and the limits of agreement (LoA) between judge pairs, we performed a Bland–Altman analysis. This method visualizes the difference between two raters against their mean, identifying the 95% limits of agreement as $\text{Mean Diff} \pm 1.96 \times SD$. The visualization and statistical summaries are presented in Fig. 8 and Table 12.

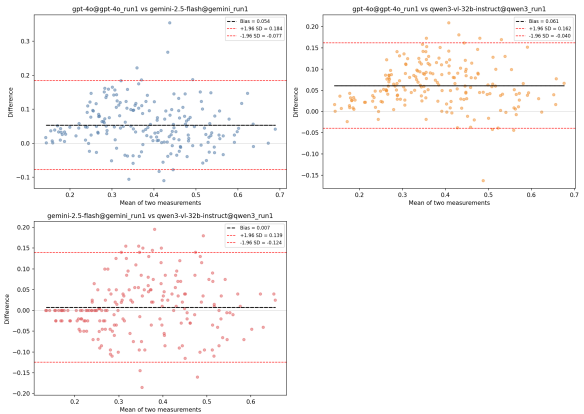


Figure 8: Bland–Altman plots illustrating the agreement between GPT-4o, Gemini-2.5-Flash, and Qwen3-VL. The dashed lines represent the 95% limits of agreement.

Table 12: Summary of Bland–Altman analysis statistics

Model comparison pair	Bias	95% LoA	n
GPT-4o vs. Gemini-2.5-Flash	+0.0536	[−0.0773, 0.1844]	200
GPT-4o vs. Qwen3-VL-32B	+0.0611	[−0.0397, 0.1619]	200
Gemini-2.5-Flash vs. Qwen3-VL-32B	+0.0075	[−0.1242, 0.1393]	200

I Multi-Agent System Prompt Templates

This appendix presents the full set of prompt templates used in the DATACROSSAGENT workflow, organized by execution stage.

I.1 Stage 1: Initial Data Profiling

Listing 2: PRELIMINARY_EVAL_PROMPT

```
Role: Data Strategy Consultant
Global Goal: {global_goal}
Current Dataset Metadata (Schema & Stats):
{data_profile}

Task:
Based ONLY on the schema and statistics provided, evaluate the
potential relevance of this dataset to the Global Goal.
Identify if this data contains core KPIs, key dimensions, or
is likely just background noise.

Output Format:
Please wrap your evaluation in the following tags:
<relevance>1-10</relevance>
<reasoning>A brief explanation</reasoning>
<priority>High/Medium/Low</priority>
```

Table 11: Transfer check on the independent 100-case UniDataBench suite.

Agent	Insight Easy	Insight Med.	Insight Hard	Insight Avg.	Summary Easy	Summary Med.	Summary Hard	Summary Avg.
ReActInsight (reported)	0.5030	0.4704	0.4517	0.4750	0.5984	0.5789	0.5605	0.5792
DataCrossAgent (ours)	0.5081	0.4687	0.4679	0.4816	0.6680	0.6224	0.6192	0.6366
Improvement (%)	+1.01	-0.36	+3.59	+1.39	+11.63	+7.51	+10.47	+9.91

I.2 Stage 2: Heuristic Exploration

Listing 3: GET_QUESTIONS_TEMPLATE

```

### Instruction:
GET_QUESTIONS_TEMPLATE = """
### Instruction:
Given the following context:
<context>{context}</context>
Given the following goal:
<goal>{goal}</goal>
Given the following schema:
<schema>{schema}</schema>

Instructions:
* Write a list of questions to be solved by the data
scientists in your team to explore my data and reach my goal.
* Explore diverse aspects of the data, and ask questions that
are relevant to my goal.
* You must ask the right questions to surface anything
interesting (trends, anomalies, etc.)
* Make sure these can realistically be answered based on the
data schema.
* The insights that your team will extract will be used to
generate a report.
* Each question should only have one part, that is a single
'?' at the end which only require a single answer.
* Do not number the questions.
* You can produce at most {max_questions} questions. Stop
generation after that.
* Most importantly, each question must be enclosed within <
question></question> tags. Refer to the example response below:

Example response:
<question>What is the average age of the customers?</question>
<question>What is the distribution of the customers based on
their age?</question>

### Response:
"""

```

Listing 4: GENERATE_CODE_SINGLE_TEMPLATE

```

GENERATE_CODE_SINGLE_TEMPLATE = """
**Goal:** {goal}
**Question:** "{question}"
**Dataset Schema:**
{schema}

**File Path:**
The dataset is located at `{database_path}`.

Instruction block begins.
**CRITICAL INSTRUCTIONS FOR WRITING PYTHON CODE:**

1.  **File Reading**:
    Requirement: You MUST load the file at `{database_path}`
    using the appropriate pandas function **based on its file
    extension** (e.g., `pd.read_csv()`, `pd.read_json()`).
    Requirement: **If reading a CSV file**: You MUST handle
    encoding errors. Use a `try-except` block. First, try `
    encoding='utf-8'`. If it fails, try `encoding='gbk'` or `
    encoding='latin1'`.
    Example for robust CSV reading:
    ```python
 import pandas as pd
 file_path = '{database_path}' # This is the path
 try:
 df = pd.read_csv(file_path, encoding='utf-8')
 except UnicodeDecodeError:
 df = pd.read_csv(file_path, encoding='gbk')
 """

```

```

...

2. **CRITICAL: Column Names Must Match the Schema Exactly**:
 Requirement: **ALWAYS** use ONLY the exact column names
 shown in the schema above.
 Requirement: **DO NOT** assume nested column names like `
 column_subfield`. The schema shows the actual column names.
 Requirement: **BEFORE** accessing any column, verify it
 exists: `if 'column_name' in df.columns:`
 Requirement: If you need data from a nested structure (e.g
 ., dict/JSON in a column), use `.apply()`:
    ```python
    # Example: extracting from a nested column
    if 'stats' in df.columns and isinstance(df['stats'].iloc
    [0], dict):
        df['total_amount'] = df['stats'].apply(lambda x: x.
        get('total_transaction_amount', 0) if isinstance(x, dict) else
        0)
    ...

    Example pattern for safe column access:
    ```python
 # List actual columns from schema
 print("Available columns:", df.columns.tolist())
 # Then access only those that exist
 if 'column_name' in df.columns:
 value = df['column_name']
 else:
 print(f"Warning: 'column_name' not found. Available:
 {{df.columns.tolist()}}")
 ...

3. **CRITICAL: Date/Time Columns**:
 Requirement: After loading the data, inspect the schema.
 If you see any columns that represent dates or times (e.g., `
 date`, `timestamp`), you **MUST** convert them to datetime
 objects using `pd.to_datetime(df['column_name'], errors='
 coerce')`.
 Requirement: **DO NOT** attempt to use string methods like
 `.strftime()` on a column before converting it to datetime.
 All date operations **MUST** use the `.dt` accessor *after*
 this conversion.
 Requirement: Use `insight.tools.safe_datetime_parse()` for
 robust date parsing if standard methods fail.

4. **Code Quality & Data Types**:
 Requirement: When creating a `pd.DataFrame` from a
 dictionary, ensure all arrays/lists have the same length to
 avoid `ValueError`.
 Requirement: Be mindful of data types. Do not assign
 string values to numeric columns or vice-versa, to avoid `
 FutureWarning`.
 Requirement: Use `insight.tools.safe_numeric_convert()`
 for converting mixed-type columns to numeric.

5. **CRITICAL: Empty DataFrame Checks**:
 Requirement: After loading data, ALWAYS check if the
 DataFrame is empty: `if df.empty: print("Warning: Empty
 DataFrame")`
 Requirement: After filtering operations, check if the
 result is empty before proceeding.
 Requirement: Before aggregations (mean, sum, etc.), verify
 there is data to aggregate.
 Example pattern:
    ```python
    filtered_df = df[df['column'] > threshold]
    if filtered_df.empty:
        print("No data matches the filter criteria")
        # Provide sensible defaults or skip the operation
    else:
        result = filtered_df['value'].mean()
    ...

```

```
6. **Error Handling**:
    Requirement: Wrap critical operations in try-except blocks.

    Requirement: For column access, verify the column exists
    first: `if 'column_name' in df.columns:`
    Requirement: Handle KeyError, ValueError, and TypeError
    gracefully.

7. **Output Generation**:
    Requirement: You **MUST** use the predefined functions
    from the `insight.tools` module to save all outputs.
    Requirement: **CRITICAL**: Chinese Font Setup*: Before
    creating any plot, you MUST call `setup()` from `insight.tools`
    to ensure Chinese characters display correctly in plots. Add
    this line before any plotting code: `setup()`.
    Requirement: Generate one simple plot and save it as a `.
    jpg` file.
    Requirement: For the plot, save a statistics summary to `
    stat.json`.
    Requirement: Save the X and Y axis data (max 50 points) to
    `x_axis.json` and `y_axis.json` respectively.
    Requirement: Each JSON file must have "name", "description
    ", and "value" fields. Ensure content is less than 4500
    characters.
    Requirement: Call `insight.tools.fix_fnames()` at the very
    end of the script.

8. **Code Structure**:
    Requirement: Start your code block with ```python and end
    it with ```.
    Requirement: Do not produce any text outside of this
    single Python code block.

**Available Tools**:
{function_docs}

Instruction block ends.
Now, write the Python code to answer the question.

```python
"""
```

Listing 5: RETRY\_TEMPLATE

```
RETRY_TEMPLATE = """You failed.

Instructions:
=====
{initial_prompt}
=====

Completion:
=====
{prev_output}
=====

Above, the Completion did not satisfy the constraints given in
the Instructions.
Error:
=====
{error}
=====

Please try again. Do not apologize. Please only respond with
an answer that satisfies the constraints laid out in the
Instructions:

"""
```

Listing 6: FOLLOW\_UP\_TEMPLATE

```
FOLLOW_UP_TEMPLATE = """
Hi, I require the services of your team to help me reach my
goal.

<context>{context}</context>

<goal>{goal}</goal>

<schema>{schema}</schema>

<question>{question}</question>

<answer>{answer}</answer>

Instructions:
* Produce a list of follow up questions explore my data and
```

```
reach my goal.
* Note that we have already answered <question> and have the
answer at <answer>, do not include a question similar to the
one above.
* Explore diverse aspects of the data, and ask questions that
are relevant to my goal.
* You must ask the right questions to surface anything
interesting (trends, anomalies, etc.)
* Make sure these can realistically be answered based on the
data schema.
* The insights that your team will extract will be used to
generate a report.
* Each question that you produce must be enclosed in <question
>content</question> tags.
* Each question should only have one part, that is a single
'?' at the end which only require a single answer.
* Do not number the questions.
* You can produce at most {max_questions} questions.

"""
```

Listing 7: SELECT\_A\_QUESTION\_TEMPLATE

```
SELECT_A_QUESTION_TEMPLATE = """
Hi, I require the services of your team to help me reach my
goal.

<context>{context}</context>

<goal>{goal}</goal>

<prev_questions>{prev_questions_formatted}</prev_questions>

<followup_questions>{followup_questions_formatted}</
followup_questions>

Instructions:
* Given a context and a goal, select one follow up question
from the above list to explore after prev_question that will
help me reach my goal.
* Do not select a question similar to the prev_questions above.

* Output only the index of the question in your response
inside <question_id></question_id> tag.
* The output questions id must be 0-indexed.

"""
```

L3 Stage 3: Key Source Identification

Listing 8: FORMAL\_ANNOTATION\_PROMPT

```
FORMAL_ANNOTATION_PROMPT = """
Role: Chief Data Scientist
Global Goal: {global_goal}
Given the following schema:
<schema>{schema}</schema>
Exploration Summary from the Agent's Deep-Dive:
{exploration_summary}

Task:
Perform a final assessment of this data's importance to the
global objective.
Metrics:
- Objective Richness S_obj (1-10): How deep, complete, and
high-quality is this source?
- Semantic Relevance S_sem (1-10): How directly does this
source support the Global Goal?
- Coordinator Utility S_LLM (1-10): How useful is this source
for downstream reasoning?
- Total Priority S_priority = 0.4*S_obj + 0.3*S_sem + 0.3*
S_LLM.

Decision Criteria:
- "Primary": Contains core metrics; can drive the main
analysis.
- "Secondary": Provides context, auxiliary dimensions, or
validation.
- Across all assessed sources, rank sources by descending
S_priority.
- Use the ranked Primary sources first before cross-
pollination.

Output Format:
<s_obj>1-10</s_obj>
<s_sem>1-10</s_sem>
<s_llm>1-10</s_llm>
```

```

<s_priority>weighted score</s_priority>
<rank>rank after sorting by S_priority</rank>
<label>Primary/Secondary</label>
<justification>Detailed reason</justification>
"""

```

```

"cross_dataset_discoveries": ["discovery 1", ...],
"limitations": ["limitation 1", ...],
"next_steps": ["recommendation 1", ...]
}}
"""

```

## I.4 Stage 4: Cross-Source Pollination

Listing 9: CROSS\_QUESTION\_PROMPT

```

CROSS_QUESTION_PROMPT = """
Role: Cross-domain Analyst
Dataset A (Your Data) Summary: {my_summary}
Dataset B (Target Data) Summary: {other_summary}
Your Label: {my_label}

Task:
Generate analytical questions that require JOINING or
COMPARING both datasets to find hidden patterns.
Constraint:
- If your label is "Primary", generate 3 deep questions.
- If your label is "Secondary", generate 1 focused question.

Output Format:
Generate your questions, each enclosed in <question> tags.
Example: <question>Your question text (Rationale: ...)</
question>
"""

```

Listing 10: ANNOTATION\_PROMPT\_TEMPLATE

```

ANNOTATION_PROMPT_TEMPLATE = """
Role: Domain Expert & Critical Reviewer

You are: {annotator_name}
Your Domain Knowledge: {annotator_knowledge}
Your Data Schema: {annotator_schema}

You are reviewing analysis results from another agent.
Target Agent: {target_agent_name}
Target Agent's Analysis Insights: {target_insight}
Target Agent's Analysis Summary: {target_summary}

Task:
Provide critical comments or cross-domain insights based on
your expertise.
Focus on:
1. Missing perspectives that your data might provide
2. Potential data quality issues
3. Alternative interpretations
4. Connections to broader business context

If you have no meaningful comments, respond with "no comment".
Otherwise, provide concise but insightful feedback.

Output Format:
<comment>Your critical feedback here</comment>
"""

```

## I.5 Stage 5: Final Summarization

Listing 11: FINAL\_PROMPT\_TEMPLATE

```

FINAL_PROMPT_TEMPLATE = """
Role: Senior Business Intelligence Analyst

Context from Complete Multi-Agent Analysis:
{full_context}

Task:
Synthesize all analyses, cross-dataset findings, and agent
annotations into a comprehensive final report.

Your output should include:
1. Executive Summary (2-3 paragraphs)
2. Key Insights (bullet points, prioritized by importance)
3. Cross-dataset Discoveries
4. Limitations and Data Quality Notes
5. Recommended Next Steps

Format your response as a JSON object with the following
structure:
{{
 "summary": "executive summary text here",
 "insights": ["insight 1", "insight 2", ...],

```

## J Cost-Quality Summary

For quick reference, Table 13 condenses the main efficiency comparison; the full token and parsing statistics remain in Table 9. Direct GPT-4o prompting minimizes latency and cost but obtains the lowest Overall score. ReActInsight reaches a higher score at substantially greater runtime and cost, whereas DataCrossAgent is close to Claude Code in quality while using about 34.9% lower estimated cost per task and 13.3% lower runtime in this profile. These estimates use the same visible evidence and scoring protocol across systems; they are intended as a reproducible reference profile rather than a claim of universal cost optimality.

Table 13: Compact cost-quality summary from the 200-task companion profile.

System	Overall	Runtime (s)	Cost/task
GPT-4o direct	0.4008	9.59	USD 0.0173
ReActInsight	0.5037	141.20	USD 0.3184
DataCrossAgent	0.5193	64.99	USD 0.1061
Claude Code	0.5188	75.00	USD 0.1630