TASKATLAS: DISCOVERING DATA-ANALYSIS TASK SPACES FROM A SIMULATED ANALYST SOCIETY

Anonymous authors
Paper under double-blind review

ABSTRACT

Constructing data-analysis benchmarks requires discovering real-world analytical tasks. Existing efforts typically rely on expert-led task-space construction, which requires continued human effort and may underrepresent analytical tasks across diverse occupations. To address these issues, we introduce TaskAtlas, a framework for discovering data-analysis task spaces from an occupationally anchored simulated analyst society. TaskAtlas selects occupations relevant to the target analytical domain from public occupational records and uses their profiles to construct simulated analysts with diverse backgrounds and work contexts. It then employs an interviewer agent to elicit their analytical needs through multi-turn conversations, extracts tasks from these needs, and organizes them into an explicit task space. We evaluate TaskAtlas in tabular text analysis and causal inference, examining the authenticity of elicited needs and the coverage of the discovered task spaces. The results show that the elicited needs receive real-world support and that the discovered task spaces recover established tasks in the reference taxonomies. More importantly, TaskAtlas enables new task formulations to emerge, including text augmentation for predictive data analysis (TA++) and latent-confounder detection (LCD), offering new directions for data-analysis research.

1 INTRODUCTION

Data-analysis benchmarks provide an essential foundation for evaluating intelligent data-analysis systems, identifying their limitations, and guiding further development (Jing et al., 2025; Chen et al., 2025; Hu et al., 2024). The tasks included in a benchmark determine which capabilities are assessed and what conclusions can be drawn. For these evaluations to reflect practical utility, the selected tasks need to correspond to analytical needs encountered in real-world workflows (Dehghani et al., 2021). These needs vary across occupations, work contexts, and decision objectives (Kandel et al., 2012), making task selection a central challenge in benchmark construction. Therefore, constructing data-analysis benchmarks requires systematically discovering analytical task spaces.

As summarized in Table 1, existing data-analysis benchmarks discover and organize tasks through practitioner surveys and interviews, expert abstraction of work scenarios, and extraction from scientific papers or user questions. These approaches ground benchmark construction in practical evidence, but extending their scope often requires additional human effort to collect, interpret, and curate task evidence. Recent work incorporates LLMs to expand expert-derived task spaces (Rahman et al., 2026), providing partial automation of this process. However, automated expansion alone does not determine whose analytical needs are represented. The representative benchmarks in Table 1 do not report explicit occupation-based sampling of need sources, leaving the resulting task spaces shaped by the practitioners, materials, and work contexts included during construction. Analytical needs arising in less represented occupational contexts may consequently be overlooked. These limitations motivate a framework that supports automated task-space expansion while systematically eliciting analytical needs across diverse occupations.

To address these limitations, our key idea is to define an explicit occupational population and use simulated interviews to elicit its analytical needs. We implement this idea in TaskAtlas, a framework that connects occupational sampling, requirement elicitation, and task-space induction. Given an analytical domain (such as tabular text analysis), TaskAtlas first uses occupational descriptions from O*NET to identify relevant occupations (National Center for O*NET Development, 2025).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Table 1: Task-space construction in existing benchmarks. "Scalability" denotes expansion beyond the initial task types. "Diversity" denotes explicit occupation-based sampling of need sources.

Benchmark	Task Source	Construction	Scalability	Diversity
DSAgentBench (Rahman et al., 2026)	Literature	LLM Induction	✓	×
LifeSciBench (Liu et al., 2026)	Scientists	Surveys	×	×
PlanningBench (Zhao et al., 2026)	Work scenarios	Expert abstraction	×	×
ScienceAgentBench (Chen et al., 2025)	Scientific papers	Task Extraction	×	×
SpreadsheetBench (Ma et al., 2024)	Forum questions	Task Extraction	×	×
InfiAgent-DABench (Hu et al., 2024)	Analysts	Interviews	×	×
TaskAtlas (ours)	Occupational records	Society simulation	✓	✓

The sampled occupational profiles anchor a simulated analyst society, specifying the responsibilities and work settings of its members. After constructing this simulated society, TaskAtlas designs an interviewer agent to probe their workflows, data information, and decision objectives through adaptive multi-turn conversations, turning occupational context into concrete analytical needs. Finally, TaskAtlas extracts analytical tasks from these needs and organizes the extracted tasks into an explicit task space.

We evaluate TaskAtlas in tabular text analysis and causal inference, examining the authenticity of elicited needs and the coverage of the discovered task spaces. To assess authenticity, we search public forums and Q&A platforms for user discussions describing the same underlying analytical problems. In tabular text analysis and causal inference, 81.6% and 68.0% of the evaluated needs receive support from at least one retrieved passage, respectively. We further compare the discovered task spaces with reference taxonomies constructed from the literature. TaskAtlas recovers all established task categories in the evaluation inventories of both domains. These results indicate that the elicited needs capture practical concerns while covering recognized analytical capabilities.

More importantly, TaskAtlas uncovers analytical needs that motivate task formulations not fully captured by the reference task space. Our evaluation highlights two examples: text augmentation for predictive data analysis (TA++) in tabular text analysis and latent-confounder detection (LCD) in causal inference. Given a table containing textual fields and a predictive objective, TA++ aims to identify useful attributes absent from the original schema and construct corresponding structured columns grounded in the available text. These attributes are inferred from the predictive objective, making schema discovery part of the task itself. LCD arises from recurring requests to determine whether two observed variables share an unobserved common cause. Its formulation requires explicit assumptions about the data-generating process and permits a not-identifiable answer when the available information is insufficient. We believe these findings open new research opportunities for the data-analysis community.

Our contributions are three-fold:

- We introduce TaskAtlas, a framework that elicits analytical needs from an occupationally anchored simulated analyst society, extracts tasks from these needs, and organizes them into explicit task spaces.
- We evaluate TaskAtlas in tabular text analysis and causal inference, demonstrating external corroboration of elicited needs and recovery of established task categories in literature-informed taxonomies.
- We identify two task formulations motivated by the elicited needs: text augmentation for predictive data analysis (TA++) and latent-confounder detection (LCD), suggesting new research directions for the data-analysis community.

2 Related Work

Data-Analysis Task-Space Construction. Discovering analytical tasks that reflect practical needs is a central challenge in constructing data-analysis benchmarks (Jing et al., 2025; Huang et al., 2024; Zhang et al., 2025a; Lai et al., 2025). Existing research approaches this challenge through several complementary sources of evidence. One line of work curates tasks from developer forums, spreadsheet user requests, and published scientific workflows (Lai et al., 2023; Ma et al., 2024; Chen et al., 2025; Gu et al., 2024). Another elicits practitioner knowledge through interviews and surveys to identify workflow categories (Hu et al., 2024; Liu et al., 2026). In the adjacent planning domain, expert abstraction organizes work scenarios into explicit task taxonomies (Zhao et al., 2026). More recently, LLMs have been incorporated into task construction to expand expert-derived taxonomies and synthesize analytical queries by composing existing tasks (Rahman et al., 2026; Qiao et al., 2026). These efforts provide practical foundations for expanding task spaces, whose scope remains shaped by the practitioners and materials included during construction. Building on this research, TaskAtlas uses an occupationally anchored simulated analyst society to elicit analytical needs, extracts tasks from these needs, and organizes them into an explicit task space.

Agent Societies. LLM-based agent societies provide a computational setting for studying how individuals with different backgrounds perceive, act, and interact in social environments. Early work such as Generative Agents equips agents with memory, reflection, and planning, showing that social behaviors can emerge through interactions among simulated individuals (Park et al., 2023). Subsequent research extends this direction to large-scale social environments that support surveys, interviews, and interventions (Piao et al., 2025). Beyond agent behavior, the construction of diverse populations grounded in real-world information has become another research focus. PersonaHub uses web-derived personas to diversify synthetic data generation (Ge et al., 2024), while other approaches use real-user profiles and distribution-based sampling to align simulated populations with target demographic characteristics or their responses with human psychometric distributions (Zhang et al., 2025b; Hu et al., 2025). These developments also support the use of simulated populations for requirements elicitation. Elicitron generates diverse user agents, simulates their product-use experiences, and conducts contextualized interviews to uncover expressed and latent needs (Ataei et al., 2025). Drawing on these ideas, TaskAtlas adopts an occupational perspective on population construction, using O*NET profiles (National Center for O*NET Development, 2025) to ground a simulated analyst society and eliciting its members' analytical needs for task-space discovery.

3 Preliminaries and Problem Formulation

3.1 Analytical Tasks and Task Spaces

An *analytical need* is a problem arising in a concrete work context, specifying what a practitioner seeks to learn or accomplish using data. We define an *analytical task* at the task-type level, i.e., a reusable specification of the input structure, analytical objective, expected output, and applicable assumptions or constraints. Needs from different occupations can support the same task type. A *task instance* instantiates a task type with concrete data and a specific query. For example, classifying free-text records is a task type and assigning categories to the records in a particular customer-feedback table is an instance.

For an analytical domain d, a *task space* is a structured collection of task types and their organizational relations:

$$\mathcal{S}_d = (\mathcal{T}_d, \mathcal{R}_d), \tag{1}$$

where $\mathcal{T}_d$ contains the task types and $\mathcal{R}_d$ specifies their organization, such as grouping by analytical objective or parent–child relations in a taxonomy. A taxonomy provides a hierarchical representation of this space.

3.2 Task-Space Discovery

Given a target domain d and public occupational records $\mathcal{O}$, *task-space discovery* aims to obtain analytical needs with real-world grounding, identify the reusable task types they express, and organize

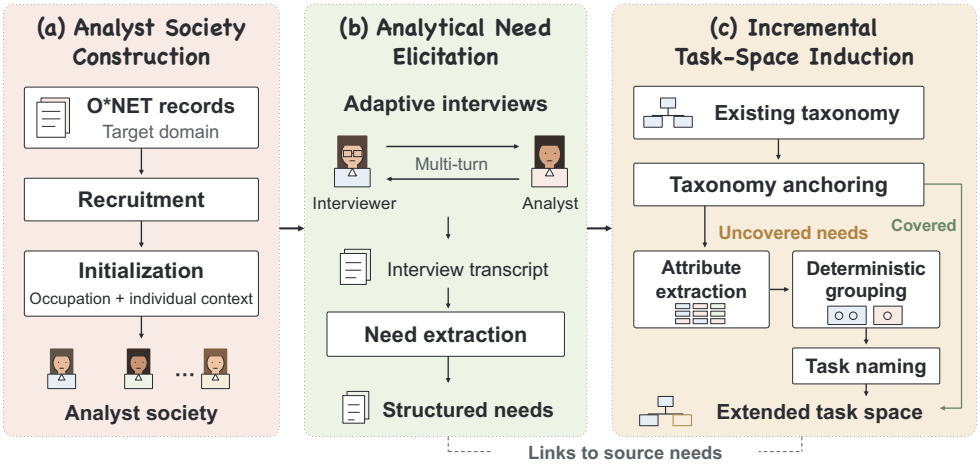

Figure 1: Overview of TaskAtlas. (a) O*NET-grounded recruitment and initialization construct a simulated analyst society. (b) Adaptive multi-turn interviews elicit analytical needs, which are extracted into structured records. (c) An existing taxonomy is extended by extracting attributes from uncovered needs, deterministically grouping them, and naming new task types. The resulting task types retain links to their source needs.

these types into an explicit task space. TaskAtlas elicits a set of needs $\mathcal{Q}_d$ from an occupationally anchored simulated analyst society and constructs

$$\mathcal{Q}_d \xrightarrow{\text{task abstraction}} \widehat{\mathcal{T}}_d \xrightarrow{\text{organization}} \widehat{\mathcal{S}}_d = (\widehat{\mathcal{T}}_d, \widehat{\mathcal{R}}_d). \tag{2}$$

The discovered types are outputs of this process. The resulting space is an empirical characterization conditioned on the population and needs examined. We assess its grounding through external evidence for the needs and its coverage through comparison with reference taxonomies. Recovering all reference categories demonstrates full coverage of the reference taxonomy.

4 TaskAtlas Framework

Practitioner interviews and expert analysis are common approaches to constructing data-analysis task spaces. To make this process scalable and broaden its coverage of analytical needs, we propose TaskAtlas, a framework for discovering data-analysis task spaces through agent-based social simulation. As shown in Figure 1, TaskAtlas comprises three stages: **Analyst Society Construction**, **Analytical Need Elicitation**, and **Incremental Task-Space Induction**. The discovered task types retain links to their source needs, making the evidence underlying the task space traceable.

4.1 Analyst Society Construction

Analyst Recruitment. Analyst recruitment determines the occupational composition of the analyst society and the work contexts from which analytical needs are elicited. To broaden the sources of analytical needs, we construct analysts with different occupational backgrounds, aiming to capture complementary perspectives and work contexts. However, manually specifying occupation lists and role descriptions requires prior knowledge of the target domain and its relevant occupations. As the scope of analytical domains and occupations expands, this process requires continued human judgment.

To improve the scalability of task-space discovery, TaskAtlas automates analyst recruitment. Given a domain ($d \in D$), we introduce a recruiter agent (M_r), prompted to assess candidate occupational backgrounds in relation to the target domain. Specifically, M_r uses O*NET occupational records, drawing on occupational descriptions and key task statements to assess the relevance of the target analytical activities to each occupation's responsibilities. It assigns a relevance score and provides a supporting rationale. The results determine the set of occupations used for analyst initialization. In

the documented recruitment run for Excel data analysis, we evaluated 1,016 candidate occupations and selected 160 occupational backgrounds spanning 14 SOC major occupational groups[1]. Table 2 summarizes the occupational composition among the selected occupations. Appendix A.2 provides the recruiter agent's full prompt and scoring rubric, and Appendix A.1 specifies the selection threshold.

Analyst Initialization. Occupational records describe practitioners' responsibilities, while task discovery requires analysts to discuss concrete tasks from their work. To connect these two levels, TaskAtlas structures each analyst profile into occupational information and individual context. Occupational information comes from O*NET, including duties, skills, and work activities, providing an occupational basis for the role. An LLM generates individual context consistent with the occupational information, including work experience, industry and organizational setting, regularly used tools, and concrete data artifacts handled at work.

Table 2: Occupational composition.

Occupational group	n (%)
Life, physical & social science	35 (21.9)
Business & finance	34 (21.3)
Management	24 (15.0)
Architecture & engineering	24 (15.0)
Office & administrative support	18 (11.3)
Computer & mathematical	13 (8.1)
Other 8 occupational groups	12 (7.5)

To initialize an analyst, we embed its complete profile in a role-playing prompt guided by two design principles. *(1) Contextual Grounding.* The role-playing prompt instructs the analyst to draw on the responsibilities and work context specified in its profile, describe needs through concrete examples involving relevant tools and data artifacts, and express analytical requests in the language it would naturally use at work. *(2) Experience Boundaries.* When asked about activities it does not perform or lacks experience with, the analyst is instructed to acknowledge these limitations rather than invent experiences. These design choices aim to ground analysts' expressed needs in concrete work contexts and keep their responses within the scope of their simulated roles' responsibilities and experience. Appendix B.1 provides an example analyst profile, and Appendix B.2 gives the role-playing prompt.

4.2 Analytical Need Elicitation

Adaptive Interviews. To elicit concrete analytical needs from analysts' work experiences, we conduct semi-structured interviews between an interviewer agent and each analyst. All analysts within a domain share the same interview protocol, with two design principles guiding the interaction. *(1) Progressive Elicitation.* The protocol moves from responsibilities and recent work episodes to current difficulties, desired assistance, and specific analytical requests. This progression provides a work context for interpreting what the analyst wants to accomplish and why. *(2) Adaptive Probing.* The interviewer generates follow-up questions based on the conversation history. When a response remains vague, it probes the data involved, the expected output, and the decision the result would support. The prompt instructs the interviewer to follow the analyst's vocabulary and introduce domain-specific probes in plain language, avoiding technical terminology unless the analyst uses it first. The interview closes with an opportunity to add needs not yet discussed. Appendix C.1 provides the interviewer prompt, and Appendix C.2 details the conversation settings.

Analytical Need Extraction. To provide a consistent input for task-space induction, an LLM extractor converts each complete interview transcript into structured need records. Each record contains an analytical request, referenced data, expected output type, usage frequency, stakes, and the intended consumer of the result. The extractor is instructed to preserve explicitly stated requests in the analyst's own words. If the analyst describes a need without stating a direct request, the extractor reformulates the description as a request while retaining the analyst's terminology. Fields that cannot be determined from the transcript are marked as unknown. Appendix D.1 provides the extraction prompt, and Appendix D.2 specifies the output schema.

[1]SOC stands for Standard Occupational Classification, the U.S. federal standard for classifying occupations based on job duties. See `https://www.bls.gov/soc/`.

4.3 Incremental Task-Space Induction

Taxonomy Anchoring. To provide a stable reference for the task space, TaskAtlas uses an existing task taxonomy as the initial structure and organizes induction as an extension of this taxonomy. Existing categories provide a common basis for interpreting requests, allowing new needs to be incorporated. Specifically, an LLM assesses whether each request can be represented by an existing category, referring to its analytical intent, user expressions, and expected outputs. The requests that are not adequately represented proceed to task-type induction. The prompt requires matches to reflect the requested analytical objective, rejecting assignments based only on shared keywords.

Task Induction. To identify analytical tasks shared across work contexts, TaskAtlas represents each uncovered need using a predefined attribute schema for the target domain. An LLM extracts attribute values from each request, and these attributes determine its group $g_d(n)$. Given the set of uncovered needs $\mathcal{U}_d$, needs with the same key form a group:

$$\mathcal{C}_{\mathbf{k}} = \{n \in \mathcal{U}_d \mid g_d(n) = \mathbf{k}\}, \tag{3}$$

where $\mathbf{k}$ is a shared combination of attribute values. This design separates semantic abstraction from deterministic grouping, establishing group membership before task definitions are generated. For each group, a naming agent uses its shared attributes and representative requests to express the common analytical objective as a task statement.

5 Experiments

To evaluate TaskAtlas as a framework for discovering data-analysis task spaces, we organize our experiments into two parts: assessing the quality of elicited analytical needs and evaluating the quality of the discovered task spaces. The results consistently demonstrate the advantages of TaskAtlas over the compared approaches. Of particular interest, the elicited needs also motivate emergent analytical tasks beyond the reference taxonomies. This section focuses on the quantitative evaluation of TaskAtlas, while Section 6 discusses these emergent tasks, their formulations, and supporting evidence in detail.

Setup. We evaluate TaskAtlas in tabular text analysis and causal inference. For the main quantitative comparisons, we consider four settings: **(1) Direct-CoT:** An LLM is prompted to reason step by step about work scenarios in the target domain and directly generate analytical needs, without personas or interviews. **(2) Single Analyst:** A single generic data-analyst agent is used without O*NET-based occupational grounding, while retaining the multi-turn interview and need-extraction procedures of TaskAtlas. **(3) Elicitron:** We adapt Elicitron (Ataei et al., 2025) by replacing product-use scenarios with data-analysis scenarios, while retaining its context-aware user generation, experience simulation, and interview-based needs elicitation. **(4) TaskAtlas:** The full framework recruits analysts with diverse occupational backgrounds from O*NET records and elicits their needs through adaptive multi-turn interviews. For task-space evaluation, all four settings use the same task-induction procedure and domain-specific reference taxonomies.

5.1 Quality of Elicited Analytical Needs

Evaluation Protocol. To assess the quality of elicited analytical needs, we evaluate their external corroboration and semantic diversity. For external evidence, we use Tavily to search public discussions on Reddit, Stack Overflow, Stack Exchange, MrExcel, Microsoft Answers, Microsoft Tech Community, Super User, and Quora. We report *Coverage@1*, the percentage of evaluated needs with at least one retrieved passage judged by GPT-4o-mini to describe or address the same underlying problem, and *Diversity*, the mean pairwise cosine distance between need-description embeddings, computed on equally sized samples across methods. These metrics assess support in public user discussions and variation in semantic content, respectively. Appendix E.1 details the evidence sources and retrieval settings, Appendix E.2 provides the judging criteria, and Appendix E.3 defines the metrics.

Results. As shown in Table 3, we observe a clear improvement in the quality of elicited analytical needs when moving from Direct-CoT to Single Analyst, Elicitron, and TaskAtlas. In both domains, TaskAtlas achieves the highest Coverage@1, indicating that the needs elicited from our simulated

Table 3: Quality of elicited analytical needs in tabular text analysis (TA) and causal inference (Causal). Coverage@1 (%) is the fraction of evaluated needs with at least one retrieved passage judged as supporting evidence. Diversity is the mean pairwise cosine distance between need-description embeddings, computed on equally sized samples.

	TA			Causal		
Method	# Needs	Coverage@1 ↑	Diversity ↑	# Needs	Coverage@1 ↑	Diversity ↑
Direct-CoT	1809	52.4%	0.70	1822	43.7%	0.65
Single Analyst	1752	62.2%	0.68	1656	54.1%	0.63
Elicitron	1528	64.8%	0.73	1368	58.3%	0.71
TaskAtlas	1333	81.6%	0.75	871	68.0%	0.74

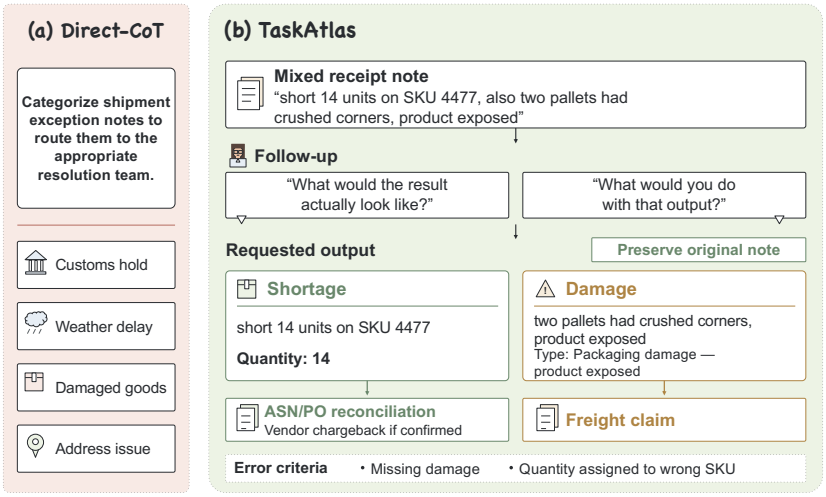

Figure 2: An example of eliciting analytical needs for logistics exception analysis.

analyst society are more frequently supported by external evidence. Meanwhile, TaskAtlas also achieves the highest semantic diversity, suggesting that it captures a broader range of analytical needs rather than concentrating on a limited set of recurring problems. Interestingly, Single Analyst improves Coverage@1 over Direct-CoT but results in lower diversity. We posit that interacting with a single analyst helps uncover more concrete and externally grounded needs, while the resulting needs remain constrained by a relatively narrow set of work contexts. In contrast, Elicitron and TaskAtlas introduce multiple analyst perspectives and achieve substantially higher diversity. In particular, grounding analyst recruitment in O*NET occupations enables TaskAtlas to cover diverse real-world work contexts while maintaining strong external corroboration.

Case Study: Logistics Exception Analysis. To complement the quantitative evaluation, Figure 2 illustrates how elicited needs specify analytical outputs and their intended uses. Panel (a) shows a Direct-CoT need to categorize shipment exception notes for routing to resolution teams. Panel (b) presents an interview with a simulated logistics analyst in consumer-electronics distribution, who describes receipt notes combining quantity shortages and packaging damage.

Follow-up questions about the desired output and its downstream use elicit more specific requirements: separate shortage and damage descriptions, a numeric shortage quantity, a damage type, and preservation of the original note. Shortage quantities support reconciliation against advance ship notices and purchase orders, with vendor chargebacks pursued if shortages are confirmed. Damage information supports freight claims. Later in the interview, the analyst identifies two error criteria: missing a damage mention and assigning a shortage quantity to the wrong SKU. This example illustrates how follow-up questions clarify the requested information, its operational use, and criteria for checking the output.

5.2 Quality of Discovered Task Space

Evaluation Protocol. We evaluate the discovered task spaces by mapping elicited needs to domain-specific reference taxonomies. For each domain, we use the same reference taxonomy and LLM-based need-to-task matching procedure across all methods. A need is mapped to a reference task type only when the type captures its analytical objective and is compatible with its expected output. We report *Task Coverage*, the percentage of reference task types supported by at least one elicited need, to measure the breadth of established analytical objectives recovered. Reference taxonomies, including category definitions and boundaries, are provided in Appendix F (TA: Appendix F.1; causal inference: Appendix F.2). Tasks beyond these taxonomies are discussed in Section 6.

Results. As shown in Table 4, TaskAtlas recovers every reference task type in both domains. The comparison with Single Analyst suggests that the value of a broader analyst population depends on the target task space: a generic role already covers the tabular text-analysis inventory, but leaves gaps in causal inference. Elicitron narrows these gaps, and TaskAtlas closes them, consistent with different work contexts contributing complementary analytical objectives. We posit that occupationally grounded recruitment provides analysts with distinct responsibilities and decision problems, helping interviews reach objectives that a single generic perspective may overlook. These findings support the use of diverse occupational contexts to broaden the sources of evidence for task-space discovery.

Table 4: Task Coverage (%) with covered/total task counts.

Method	**TA** ↑	**Causal** ↑
Direct-CoT	83.3 (10/12)	76.9 (10/13)
Single Analyst	100.0 (12/12)	76.9 (10/13)
Elicitron	100.0 (12/12)	92.3 (12/13)
TaskAtlas	100.0 (12/12)	100.0 (13/13)

6 Emergent Tasks

Beyond recovering reference taxonomies, we examine elicited needs and their source transcripts for requirements that suggest task-space extensions. This analysis motivates two candidate *emergent tasks*: *Text Augmentation for Predictive Data Analysis* and *Latent-Confounder Detection*. Figure 3 illustrates their inputs and intended outputs.

6.1 Text Augmentation for Predictive Data Analysis

Some elicited needs specify a downstream objective without specifying which textual attributes would support it. A simulated environmental restoration planner, for example, asks whether initial site-assessment narratives predict later restoration outcomes, leaving the relevant narrative characteristics open.

Figure 3(a) illustrates this formulation with restaurant reviews. Given an objective of improving satisfaction, the system discovers attributes such as service quality, food quality, and waiting time, then derives their values for subsequent prediction. The user supplies the objective without prescribing these columns; unmentioned aspects remain unknown.

We formulate this task as

$$(X, g, C) \longmapsto (S, A, [X; A]), \tag{4}$$

where X is the original table containing textual fields, g is the downstream objective, and C is optional authorized context. The output comprises a discovered attribute schema S, its values A, and the table $[X; A]$ formed by appending the new columns to X.

TA++ combines T4 (open-schema extraction) and T12 (column generation), with the predictive objective guiding schema discovery (Appendix F.1). An unspecified schema alone does not establish an uncovered task. Attributes and values must remain grounded in X or C.

6.2 Latent-Confounder Detection

A second pattern concerns requests to determine whether an observed association reflects a shared cause. Our review of 479 causal-inference transcripts identifies eight confounding-related examples,

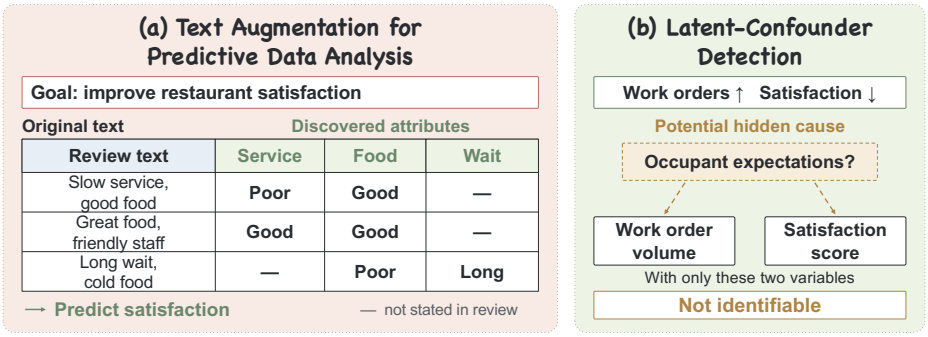

Figure 3: Examples of candidate emergent tasks. (a) A restaurant-improvement objective guides attribute discovery from reviews for satisfaction prediction. (b) Work-order volume and satisfaction may share an unobserved cause, requiring further assumptions or evidence for identification. Table entries are illustrative; dashed arrows denote a hypothesis.

including hypothesized common causes and requests to reveal confounding. These needs motivate a more specific task: assessing whether two observed variables share an unobserved cause, an objective not explicitly represented in our reference causal taxonomy.

Figure 3(b) draws on a simulated facilities manager's account of buildings with more work orders and lower satisfaction. The analyst asks whether work-order volume drives dissatisfaction or whether occupant expectations influence both measures, and what additional data could resolve the question. The answer would inform whether faster maintenance is likely to improve satisfaction.

We formulate this task as

$$(\mathcal{D}_{XY}, \mathcal{A}) \longmapsto (\hat{z}, \hat{c}, \hat{e}), \tag{5}$$

where $\mathcal{D}_{XY}$ contains observations of two variables and $\mathcal{A}$ specifies the data-generating assumptions under which the question is assessed. The decision $\hat{z} \in \{\text{YES}, \text{NO}, \text{NOT-IDENTIFIABLE}\}$ indicates whether a shared latent cause can be established, ruled out, or left unresolved under those assumptions. The outputs $\hat{c}$ and $\hat{e}$ provide a confidence assessment and a diagnostic explanation, respectively.

Identifiability is part of the task specification: two observed variables alone do not generally determine whether they share a latent cause. In the figure, occupant expectations remain a hypothesis, making NOT-IDENTIFIABLE a valid response without further assumptions or evidence. A lack of evidence for confounding does not by itself justify a NO decision.

These examples connect concrete work problems to two analytical requirements: discovering a representation suited to an objective, and assessing a possible hidden explanation for an association. Formalizing these requirements provides a basis for refining candidate tasks beyond the reference taxonomies.

7 Conclusion

We present TaskAtlas, a framework for discovering data-analysis task spaces through interviews with occupationally grounded simulated analysts. By combining occupational context, adaptive elicitation, and incremental task-space induction, the framework connects concrete work needs to explicit task types while preserving links to their source evidence. Across tabular text analysis and causal inference, TaskAtlas elicits needs with stronger external corroboration and greater semantic diversity than the evaluated baselines, while covering all reference task types. The elicited needs also motivate candidate tasks beyond these taxonomies, including text augmentation for predictive data analysis and latent-confounder detection. These findings suggest that simulated interviews can complement expert-led task-space construction, broaden its occupational scope, and inform future data-analysis benchmarks.

AI USE STATEMENT

In this work, generative AI models support occupational relevance assessment, analyst-profile generation, simulated interviews, analytical-need extraction, task-space induction, and evidence judging. These uses produce the synthetic profiles, interviews, and need records described in the paper. We also used generative AI tools to polish the manuscript and assist with literature retrieval and discovery. We did not use these tools to draft substantive sections of the manuscript or prove mathematical claims.

ETHICS STATEMENT

This work uses publicly available occupational information and online community data, and involves no human subjects or private personal information. The simulated analyst profiles are constructed from occupational information rather than real individuals. The intended application is to support the discovery and evaluation of data-analysis tasks.

REPRODUCIBILITY STATEMENT

The framework and experimental setup are described in Sections 4 and 5. Appendices A–D document analyst recruitment and initialization, interviewer prompts and conversation settings, and the extraction prompt and output schema. Appendix E details the evidence sources, judging criteria, and metric definitions. Appendix F presents the reference taxonomies and category boundaries.

REFERENCES

Mohammadmehdi Ataei, Hyunmin Cheong, Daniele Grandi, Ye Wang, Nigel Morris, and Alexander Tessier. Elicitron: A large language model agent-based simulation framework for design requirements elicitation. *Journal of Computing and Information Science in Engineering*, 25(2):021012, 2025.

Ziru Chen, Shijie Chen, Yuting Ning, Qianheng Zhang, Boshi Wang, Botao Yu, Yifei Li, Zeyi Liao, Chen Wei, Zitong Lu, et al. Scienceagentbench: Toward rigorous assessment of language agents for data-driven scientific discovery. In *International Conference on Learning Representations*, volume 2025, pp. 96934–96990, 2025.

Mostafa Dehghani, Yi Tay, Alexey A Gritsenko, Zhe Zhao, Neil Houlsby, Fernando Diaz, Donald Metzler, and Oriol Vinyals. The benchmark lottery. *arXiv preprint arXiv:2107.07002*, 2021.

Tao Ge, Xin Chan, Xiaoyang Wang, Dian Yu, Haitao Mi, and Dong Yu. Scaling synthetic data creation with 1,000,000,000 personas. *arXiv preprint arXiv:2406.20094*, 2024.

Ken Gu, Ruoxi Shang, Ruien Jiang, Keying Kuang, Richard-John Lin, Donghe Lyu, Yue Mao, Youran Pan, Teng Wu, Jiaqian Yu, et al. Blade: Benchmarking language model agents for data-driven science. In *EMNLP*, pp. 13936–13971, 2024.

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. Infiagent-dabench: evaluating agents on data analysis tasks. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

Zhengyu Hu, Jianxun Lian, Zheyuan Xiao, Max Xiong, Yuxuan Lei, Tianfu Wang, Kaize Ding, Ziang Xiao, Nicholas Jing Yuan, and Xing Xie. Population-aligned persona generation for llm-based social simulation. *arXiv preprint arXiv:2509.10127*, 2025.

Yiming Huang, Jianwen Luo, Yan Yu, Yitong Zhang, Fangyu Lei, Yifan Wei, Shizhu He, Lifu Huang, Xiao Liu, Jun Zhao, et al. Da-code: Agent data science code generation benchmark for large language models. In *EMNLP*, pp. 13487–13521, 2024.

Liqiang Jing, Zhehui Huang, Xiaoyang Wang, Wenlin Yao, Wenhao Yu, Kaixin Ma, Hongming Zhang, Xinya Du, and Dong Yu. Dsbench: How far are data science agents from becoming data

science experts? In *International Conference on Learning Representations*, volume 2025, pp. 32597–32649, 2025.

Sean Kandel, Andreas Paepcke, Joseph M Hellerstein, and Jeffrey Heer. Enterprise data analysis and visualization: An interview study. *IEEE transactions on visualization and computer graphics*, 18 (12):2917–2926, 2012.

Eugenie Lai, Gerardo Vitagliano, Ziyu Zhang, Om Chabra, Sivaprasad Sudhir, Anna Zeng, Anton A Zabreyko, Chenning Li, Ferdi Kossmann, Jialin Ding, et al. Kramabench: A benchmark for ai systems on data-to-insight pipelines over data lakes. *arXiv preprint arXiv:2506.06541*, 2025.

Yuhang Lai, Chengxi Li, Yiming Wang, Tianyi Zhang, Ruiqi Zhong, Luke Zettlemoyer, Wen-tau Yih, Daniel Fried, Sida Wang, and Tao Yu. Ds-1000: A natural and reliable benchmark for data science code generation. In *ICML*, pp. 18319–18345, 2023.

Amelia Liu, Andrew Ho, Anne Marie Droste, David Martin, Edmund Wong, Edward Zhou, Isabelle Zhou, Joshua Park, Joy Jiao, Katie-Rose Skelly, et al. Lifescibench: Evaluating language models on realistic, expert-level tasks in the life sciences. *bioRxiv*, pp. 2026–08, 2026.

Zeyao Ma, Bohan Zhang, Jing Zhang, Jifan Yu, Xiaokang Zhang, Xiaohan Zhang, Sijia Luo, Xi Wang, and Jie Tang. Spreadsheetbench: Towards challenging real world spreadsheet manipulation. *Advances in Neural Information Processing Systems*, 37:94871–94908, 2024.

National Center for O*NET Development. O*NET database. `https://www.onetcenter.org/database.html`, 2025. Accessed 2026-07-27.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pp. 1–22, 2023.

Liana Patel, Siddharth Jha, Melissa Pan, Harshit Gupta, Parth Asawa, Carlos Guestrin, and Matei Zaharia. Semantic operators: A declarative model for rich, ai-based data processing, 2025. URL `https://arxiv.org/abs/2407.11418`.

Jean-Philippe Pellet and André Elisseeff. Using markov blankets for causal structure learning. *Journal of Machine Learning Research*, 9(43):1295–1342, 2008. URL `http://jmlr.org/papers/v9/pellet08a.html`.

Jinghua Piao, Yuwei Yan, Jun Zhang, Nian Li, Junbo Yan, Xiaochong Lan, Zhihong Lu, Zhiheng Zheng, Jing Yi Wang, Di Zhou, et al. Agentsociety: Large-scale simulation of llm-driven generative agents advances understanding of human behaviors and society. 2025.

Shuofei Qiao, Yanqiu Zhao, Zhisong Qiu, Xiaobin Wang, Jintian Zhang, Zhao Bin, Ningyu Zhang, Yong Jiang, Pengjun Xie, Fei Huang, et al. Scaling generalist data-analytic agents. In *International Conference on Learning Representations*, volume 2026, pp. 14942–14971, 2026.

Mizanur Rahman, Mohammed Saidul Islam, Ridwan Mahbub, Md Tahmid Rahman Laskar, Shafiq Joty, and Enamul Hoque Prince. Dsagentbench: Can agents automate end-to-end data-science workflows in real computer environments? *arXiv preprint arXiv:2608.10366*, 2026.

Shreya Shankar, Tristan Chambers, Tarak Shah, Aditya G. Parameswaran, and Eugene Wu. Docetl: Agentic query rewriting and evaluation for complex document processing, 2025. URL `https://arxiv.org/abs/2410.12189`.

Ali Shojaie and Emily B. Fox. Granger causality: A review and recent advances, 2021. URL `https://arxiv.org/abs/2105.02675`.

Xinyue Wang, Kun Zhou, Wenyi Wu, Har Simrat Singh, Fang Nan, Songyao Jin, Aryan Philip, Saloni Patnaik, Hou Zhu, Shivam Singh, Parjanya Prashant, Qian Shen, and Biwei Huang. Causal-copilot: An autonomous causal analysis agent, 2025. URL `https://arxiv.org/abs/2504.13263`.

Dan Zhang, Sining Zhoubian, Min Cai, Fengzu Li, Lekang Yang, Wei Wang, Tianjiao Dong, Ziniu Hu, Jie Tang, and Yisong Yue. Datascibench: An llm agent benchmark for data science. *arXiv preprint arXiv:2502.13897*, 2025a.

Xinnong Zhang, Jiayu Lin, Xinyi Mou, Shiyue Yang, Xiawei Liu, Libo Sun, Hanjia Lyu, Yihang Yang, Weihong Qi, Yue Chen, et al. Socioverse: A world model for social simulation powered by llm agents and a pool of 10 million real-world users. *arXiv preprint arXiv:2504.10157*, 2025b.

Yuwei Zhang, Zihan Wang, and Jingbo Shang. Clusterllm: Large language models as a guide for text clustering, 2023. URL https://arxiv.org/abs/2305.14871.

Ziliang Zhao, Zenan Xu, Shuting Wang, Hongjin Qian, Yan Lei, Minda Hu, Zhao Wang, Shihan Dou, Zhicheng Dou, and Pluto Zhou. Planningbench: Generating scalable and verifiable planning data for evaluating and training large language models. *arXiv preprint arXiv:2605.20873*, 2026.

A Analyst Recruitment

This appendix documents the recruitment run that produced the 160 occupational backgrounds in Table 2. The saved run screens occupations for Excel data analysis, using O*NET occupational descriptions and task statements. The prompt is parameterized by an analytical activity; the configuration below reports the activity and threshold recorded for this run.

A.1 Recruitment Setup and Selection Threshold

Occupational inputs. The recruiter receives a compact card for each candidate occupation: its title, SOC code, occupational description, up to six task statements with the highest O*NET importance scores, and a tools field. Each task statement retains its O*NET task identifier. Descriptions longer than 280 characters are truncated and followed by an ellipsis. The implementation groups cards into batches of 20 by default and requests a score, supporting evidence, and a one-sentence rationale for each candidate.

Saved recruitment configuration

Screening activity	Excel data analysis (English translation of the saved Chinese input)
Recruiter model	`claude-opus-4.6`
Candidate occupations	1,016
Scoring scale	Integer scores from 0 to 5
Selection threshold	Score ≥ 2
Output size	All occupations meeting the threshold; no fixed top-k limit
Selected occupations	160, spanning 14 SOC major groups

Selection rule. After scoring, the implementation removes candidates below the threshold and deduplicates repeated occupation titles, retaining the highest score for each title. It sorts the remaining occupations by score. The saved run sets `top=ALL`, so all survivors are retained and the optional LLM reranking step is skipped. The selected set contains 110 occupations with score 2, 37 with score 3, and 13 with score 4; none has score 5. Under the rubric in Appendix A.2, the threshold admits occupations with peripheral or occasional relevance, rather than requiring data analysis to be a core duty. These scores are model judgments of occupational relevance, not measurements of workers' task frequency.

Implementation details. The scoring call in `select_occup.py` uses temperature 0 and a maximum output of 4,000 tokens. Although the prompt permits evidence from tasks or tools, the current card builder produces empty tools lists for the supplied records: it expects nested `examples`, whereas the records store tool names directly. Accordingly, the saved selection should not be interpreted as enforcing a tool-use requirement. The 160 selected SOC codes match the occupational records saved for analyst initialization.

A.2 Recruiter Prompt and Scoring Rubric

The following templates reproduce the scoring prompt in `select_occup.py`, with formatting adjusted for readability. The placeholder {`task`} receives the screening activity, and {`cards`} receives the batch of occupational cards serialized as JSON. The six score definitions below are the rubric supplied to the recruiter.

System prompt: occupation recruiter

You evaluate which O*NET occupations realistically perform a given task. Be strict: a high score means the task is a routine, central part of the occupation's work, supported by the listed tasks/tools. Reply ONLY with valid JSON, no commentary.

User-message template: occupation scoring

`TASK_DESCRIPTION:`
{`task`}

CANDIDATE OCCUPATIONS (this batch):
{`cards`}

Score every candidate on a 0–5 scale for how realistically a typical worker in that occupation performs the task above as part of their normal job:

5 = core duty, explicitly evidenced by listed tasks/tools
4 = common duty, strongly implied
3 = sometimes done
2 = rarely / tangential
1 = unlikely
0 = no real connection

For each candidate, cite evidence: `task_id(s)` from "`top_tasks`" or tool name(s) from "`tools`" that support the score. Empty list if score ≤ 1.

Return JSON in EXACTLY this schema (no extra keys, no prose):

```
{
  "results": [
    {"title": "<exact title>", "score": <int 0-5>,
     "evidence": ["<task_id or tool name>", ...],
     "reason": "<one short sentence>"}
  ]
}
```

The implementation attaches the SOC code to each returned record by matching its title to the input card. The saved output retains the score, evidence, and rationale for each selected occupation.

B Analyst Initialization

An analyst profile combines an O*NET occupational record with LLM-generated individual context. The implementation retains the occupational record in an `occupation` field and adds a simulated identity, work experience, organizational setting, and an `individual_variance` block. The complete profile is serialized as JSON and inserted into the analyst's system prompt.

B.1 Example Analyst Profile

The example below is condensed from the saved profile `P_19_2041_02`. Occupational lists are shortened for readability; the runtime prompt receives the complete record. The individual context, including personality traits and initial workflow difficulties, is generated for the simulation and is not evidence collected from a human participant.

Example profile: occupational information from O*NET

Occupation: Environmental Restoration Planners **SOC:** 19-2041.02

Description. Collaborate with field and biology staff to oversee the implementation of restoration projects and to develop new products. Process and synthesize complex scientific data into practical strategies for restoration, monitoring or management.

Selected duties.

- Develop environmental restoration project schedules and budgets.
- Provide technical direction on environmental planning to energy engineers, biologists, geologists, or other professionals working to develop restoration plans or strategies.
- Create habitat management or restoration plans, such as native tree restoration and weed control.

Selected skills: Critical Thinking; Reading Comprehension; Active Listening.

Selected work activities: Getting Information; Working with Computers; Communicating with Supervisors, Peers, or Subordinates.

Example profile: LLM-generated individual context

Name: Marcus Tran-Villanueva **Experience:** 15 years
Age / **gender:** 43 / male
Domain: Federal Government—Fish and Wildlife
Organization: Cascadia Watershed Collaborative; small (50–200 employees).

Regularly used tools. ESRI ArcGIS software; Microsoft Excel; HEC-RAS; Autodesk AutoCAD Civil 3D; Microsoft Project.

Data artifacts. Site-assessment narratives describing habitat conditions, invasive species, and erosion; permit justifications referencing regulatory requirements; grant scope-of-work descriptions and budget rationales; remediation-plan deviation notices explaining changes in the field.

Interview-related traits. Thinks aloud and revises his reasoning; checks GIS data quality rather than trusting partner-agency layers; is patient with stakeholder disagreements but frustrated by permit delays.

Initial workflow difficulties.

- Field crews describe similar habitat conditions using different terminology, complicating aggregation into a prioritization spreadsheet.
- Reusing permit narratives across jurisdictions requires manually revising regulatory terminology and statute references.
- Funding agencies interpret the same restoration activity differently, making grant scope descriptions difficult to reconcile.

B.2 Analyst Role-Playing Prompt

The following system-prompt template reproduces the wording of the analyst initialization function, with line wrapping and list formatting adjusted for presentation. The placeholder {persona_block} is replaced by the complete profile serialized using `json.dumps(persona, indent=2, ensure_ascii=False)`. The instructions explicitly connect responses to the profile and require the analyst to acknowledge tasks outside its experience.

System prompt: analyst role-playing

You are role-playing as the following person in a user research interview. Stay fully in character throughout the conversation. Respond naturally, as a real person would in a professional interview setting.

```
=== YOUR PERSONA ===
{persona_block}
=== END PERSONA ===
```

Guidelines for your responses:

- Draw from your persona's specific domain, role, and workflow
- Use domain-specific vocabulary and examples from your field
- Be specific: mention actual tools, data types, and scenarios from your work
- Vary your response depth: sometimes give detailed examples, sometimes be briefer
- Show your personality traits naturally (don't announce them)
- If asked about tasks you don't do, say so honestly—don't make things up to please the interviewer
- When describing queries you'd want, phrase them as you naturally would—not as a formal specification
- It's OK to think out loud, pause, or revise your answer

Respond as this person would in a real interview. Output ONLY your response—no labels, no meta-commentary.

The same system prompt is supplied at each analyst turn. The conversation context, ending with the interviewer's latest question, is passed separately in the user message:

User-message template: analyst turn

Here is the conversation so far:

{conversation_context}

The interviewer just spoke. Respond naturally as your persona.

Implementation correspondence. The occupational record is retained by the profile-merging function in `Simulation/generate_personas.py`. The system prompt and per-turn user message are constructed in `Simulation/interview.py` by `_candidate_system_prompt` and `_build_conversation_context`, respectively. These details specify the analyst's initialization; the interviewer protocol is supplied separately.

C Adaptive Interviews

This appendix provides the interviewer prompt and conversation settings from the simulation implementation. Each interview alternates between an interviewer and a simulated analyst initialized as described in Appendix B.

C.1 INTERVIEWER PROMPT

The following system-prompt template reproduces `_interviewer_system_prompt` in `Simulation/interview.py`. The placeholder {`protocol`} receives the domain's interview protocol. The shared wrapper retains text-analysis wording and a Q9 reference, including when a causal protocol is inserted; these are reproduced unchanged below.

System prompt: adaptive interviewer

You are an experienced UX researcher conducting a semi-structured interview for a study on how professionals work with text data in spreadsheets and tabular databases.

Your interview protocol is below. Follow it as a guide, but adapt your follow-up questions based on what the participant says. Good interviewing means:

- Listen actively and probe on specifics ("Can you give me a concrete example?")
- When the participant mentions a task, ask what data it involves and what the expected output looks like
- Ask "why"—what business need drives this task?
- Don't lead the participant toward specific answers
- If they mention something interesting, follow that thread even if it departs from the protocol
- Use natural transitions between topics
- Be warm and professional

CRITICAL DISCIPLINES (these are non-negotiable):

1. VOCABULARY DISCIPLINE—In Sections 1 and 2 (warm-up and context), do not introduce the words "spreadsheet," "column," "table," or "dataset" before the participant uses them. Refer to "a recent piece of work" or "the information you handle." The protocol's Q3 explicitly names "text column in a spreadsheet or table"—that is the FIRST point at which you may use those terms. The point is to let the participant supply the noun for their own work; if you supply it first, you contaminate their answer.

2. Q9 VERBATIM CAPTURE—When the participant articulates plain-English instructions in Q9 (Section 4), capture them word-for-word. Do not smooth grammar, complete fragments, or paraphrase. The participant's exact wording is primary data for this study.

3. NO TECHNICAL VOCABULARY—Do not introduce terms like "classification," "sentiment analysis," "entity extraction," "NER," or "semantic matching" unless the participant uses them first.

```
=== INTERVIEW PROTOCOL ===
{protocol}
=== END PROTOCOL ===
```

You are conducting this interview turn by turn. Each time you speak, output ONLY your next interviewer statement or question—nothing else. Do not include labels like "Interviewer:"—just speak naturally.

C.2 Conversation Settings

The loop sets a target of $\lfloor (8+12)/2 \rfloor = 10$ main question–answer exchanges, followed by a closing question and one final analyst response. Completed transcripts therefore contain 11 exchanges (22 messages), consistent with the inspected saved runs. The values 8 and 12 are configuration bounds used to compute the target, not a randomly sampled interview length.

Context and generation settings

Dialogue context	Up to 10 messages: the full history initially; thereafter the first 2 and latest 8 messages, with an omission marker.
Role context	The corresponding system prompt is supplied at every turn; the analyst's complete profile remains in its system prompt.
Current client settings	Temperature 0.7; maximum output 32,768 tokens per call, shared by interviewer and analyst.

These sampling settings describe the current `Simulation/llm` client. Archived causal runs record `claude-opus-4.7`; the archived TA run records only the Copilot backend default, without a resolved model identifier. The current client's sampling settings are not independently verified for every historical run.

Interviewer messages. The following user-message templates control opening, follow-up, and closure. Here, {`conversation_context`} contains the visible, speaker-labeled history. The closing template appends its instruction to the complete follow-up message, denoted by {`interviewer_context`}.

User message: opening

Begin the interview. Greet the participant warmly, introduce yourself, and briefly set the scope using the protocol's phrasing: you are researching how people in different jobs work with the information they handle every day. Then ask Q1 (the first warm-up question). Do NOT mention 'spreadsheets,' 'tables,' 'data,' or 'columns' in this opening—those terms are introduced only when the participant uses them first or when the protocol reaches Q3.

User-message template: follow-up

Here is the conversation so far:
{`conversation_context`}
Based on the conversation, ask your next question or follow-up. If you've covered all protocol sections and have enough material (at least 8 exchanges), you may begin wrapping up the interview.

User-message template: closing

{`interviewer_context`}
It's time to wrap up. Thank the participant, ask if there's anything else they'd like to add about working with text data in spreadsheets that you haven't covered, and close the interview warmly.

D Analytical Need Extraction

After an interview, the extractor receives the complete transcript with speaker labels and the analyst's name, role, organization, and identifier. Extraction uses the full transcript, without the conversation window applied during interviewing. The implementation submits the following template as a user message, without a separate extraction system prompt.

D.1 Extraction Prompt

The template reproduces `_build_query_extraction_prompt` in `Simulation/interview.py`. Placeholders indicate the inserted metadata and transcript; the implementation's text-analysis wording and question references are retained.

User-message template: analytical need extraction

You are a research analyst extracting structured records from an interview transcript. The participant is {name} ({role} at {company}).

```
=== TRANSCRIPT ===
{transcript_text}
=== END TRANSCRIPT ===
```

Extract every text-analysis instruction the participant articulated, with special attention to Section 4 (Q9—Query Articulation), which is the primary source. For each instruction, produce the v2 six-field demand record using the EXACT enums listed.

Per-instruction record:

- "query_id": sequential, format "{candidate_id}-Q1", "{candidate_id}-Q2", ...
- "verbatim_instruction": the EXACT words the participant used, as they would type the instruction. Do not paraphrase, complete, or smooth grammar. If the participant described the task without phrasing it as an instruction, reconstruct the most faithful instruction form using their own vocabulary.
- "data_referenced": which spreadsheet/table this instruction targets, and which columns are relevant—1–2 sentences, in the participant's vocabulary.
- "expected_output": one of: "new_column" | "summary" | "filtered_list" | "number" | "report" | "other"
- "frequency": one of: "one_shot" | "per_event" | "weekly" | "monthly" | "ad_hoc"
- "stakes": one of: "low" | "medium" | "high"
- "stakes_rationale": ONE sentence describing what goes wrong if the result is wrong (the participant's own framing).
- "consumer": one of: "self" | "team" | "cross_functional" | "decision_body"
- "supporting_quote": ONE verbatim quote from the transcript that establishes this instruction.

If a field cannot be determined from the transcript, use "unknown" (string)—do NOT invent a value. The Phase-4 demand-weight aggregation handles "unknown" gracefully by assigning a neutral weight.

Also extract these aggregate fields:

- "novel_observations": observations about text-analysis needs that don't fit cleanly into a single instruction (list of strings; may be empty).
- "participant_priorities": if the participant ranked or prioritized tasks in Q10, return an ordered list of query_ids (most important first); otherwise null.

Output a single JSON object with:

- "candidate_id": "{candidate_id}"
- "queries": [list of instruction records]
- "novel_observations": [list of strings]
- "participant_priorities": [ordered list of query_ids] or null

Output ONLY the JSON object. No markdown, no preamble.

D.2 Output Schema

Each element of `queries` is a structured need record. Its six core fields correspond to the request, referenced data, output type, frequency, stakes, and consumer described in Section 4.2. Three additional fields retain a record identifier, a rationale for the stakes, and supporting evidence.

Need-record fields

Field	Content
`verbatim_instruction`	Explicit request in the analyst's words, or a faithful instruction reconstructed from a described need.
`data_referenced`	Relevant data artifact and columns, described in the analyst's vocabulary.
`expected_output`	Output category from the prompt's enumeration.
`frequency`	Usage frequency from the prompt's enumeration.
`stakes`	Consequence level: low, medium, or high.
`consumer`	Intended recipient: self, team, cross-functional stakeholders, or a decision body.
`query_id`	Analyst identifier followed by a sequential suffix, such as `-Q1`.
`stakes_rationale`	One sentence explaining the consequence of an incorrect result.
`supporting_quote`	A transcript excerpt supporting the need.

The enclosing object also records `candidate_id`, a list of `novel_observations`, and `participant_priorities`. The latter is an ordered list of need identifiers when a ranking is stated, and JSON `null` otherwise. The layout below uses placeholders for transcript-dependent strings and `"unknown"` to illustrate missing categorical values.

JSON output layout

```
{
  "candidate_id": "{candidate_id}",
  "queries": [
    {
      "query_id": "{candidate_id}-Q1",
      "verbatim_instruction": "{request}",
      "data_referenced": "{data_description}",
      "expected_output": "unknown",
      "frequency": "unknown",
      "stakes": "unknown",
      "stakes_rationale": "{stakes_rationale}",
      "consumer": "unknown",
      "supporting_quote": "{transcript_excerpt}"
    }
  ],
  "novel_observations": [],
  "participant_priorities": null
}
```

Evidence and missing values. The prompt permits reconstructing a request when the analyst describes a need without stating an instruction. Accordingly, `verbatim_instruction` is not always a literal transcript substring; `supporting_quote` is requested separately as verbatim evidence. The analyst identifier and sequential need identifier link each record to its source interview. Undetermined record fields receive the string `"unknown"`. These are prompt-level requirements: the implementation parses JSON but does not enforce field-level types or enumerations.

E Evaluation of Elicited Analytical Needs

This appendix details the evidence-retrieval and judging procedures, followed by the definitions of external corroboration and semantic diversity.

E.1 Evidence Sources and Retrieval

The retrieval implementation uses Tavily with the following domains supplied as inclusion filters. It requests basic search, disables generated answers and raw page content, and retains the returned titles, URLs, snippets, relevance scores, and search queries. Thus, the evidence judged below consists of retrieved snippets.

Configured evidence sources

Community	Domain filter
Reddit	`reddit.com`
Stack Overflow	`stackoverflow.com`
Stack Exchange	`stackexchange.com`
MrExcel	`mrexcel.com`
Microsoft Answers	`answers.microsoft.com`
Microsoft Tech Community	`techcommunity.microsoft.com`
Super User	`superuser.com`
Quora	`quora.com`

Query construction. The query-rewriting mode requests one query of 5–10 words for each need, removing organization-specific details while retaining the underlying problem. The default rewriting model is Claude Haiku 4.5, with temperature 0 and a maximum output of 300 tokens. Query strings are capped at 380 characters. The retrieval stage removes duplicate URL strings, sorts results by Tavily score, and retains at most five results per need.

System prompt: search-query rewriting

You convert a worker's pain point description into short web-search queries that practitioners would use when posting about the same problem on forums like Reddit, Stack Overflow, or MrExcel. Strip company/role-specific nouns. Reply ONLY with valid JSON.

User-message template: search-query rewriting

PAIN POINT:
– role: {role}
– description: {description}
– data_involved: {data_involved}

Generate 1 short search query (5–10 words) that someone with this exact problem would plausibly type into a search engine, or that a forum post title would contain. Use practitioner vocabulary, not the worker's narrative phrasing. Strip company-specific nouns (e.g. "six supervisor emails" → "extracting numbers from emails").

Return JSON in EXACTLY this schema (no extra keys, no prose):

```
{
  "queries": ["<query>"]
}
```

E.2 Evidence-Judging Criteria and Prompt

GPT-4o-mini judges each need–snippet pair using the templates below. The call uses temperature 0, a maximum output of 200 tokens, and the first 1,500 characters of each snippet. Unavailable contextual fields are supplied as empty strings. Only the `support` label contributes to Coverage@k; related topics and shared keywords alone are insufficient.

System prompt: evidence judging

You are an evidence judge. Given a pain point voiced by a worker and a retrieved web chunk, decide whether the chunk is evidence that practitioners actually experience this kind of problem. IMPORTANT: a forum post asking how to solve a problem IS evidence that the problem exists—you do not need the chunk to literally complain. A tool / workaround / solution discussion for the same problem also counts as evidence. What matters is the underlying problem, not the tone or framing. Reply ONLY with valid JSON.

User-message template: evidence judging

PAIN POINT (voiced by a worker):
– role: {role}
– description: {description}
– supporting_quote: {quote}
– data_involved: {data_involved}

RETRIEVED CHUNK (from a public practitioner forum):
– url: {url}
– title: {title}
– snippet: {snippet}

LABELS (judge by the underlying PROBLEM, not by surface phrasing):

- "support": the chunk is clear evidence that practitioners encounter the same kind of problem. This includes: someone describing the problem, someone asking how to solve it, someone proposing a tool/workaround for it, or someone comparing approaches to it. The problem area must match—e.g. "extracting structured data from email narratives" supports a pain point about manually parsing email reports, even if one says "automate" and the other says "tedious".
- "partial": the chunk is in an adjacent problem area but is clearly a different specific problem (e.g. same tool but a different feature, same domain but a different workflow stage).
- "unrelated": off-topic, irrelevant, or only superficially keyword-matched.

Return JSON in EXACTLY this schema (no extra keys, no prose):

```
{
  "label": "<support | partial | unrelated>",
  "reason": "<one short sentence justifying the label>"
}
```

E.3 METRIC DEFINITIONS

External corroboration: Coverage@k. Let $\mathcal{N}_{\text{eval}}$ be the set of evaluated need records and $\mathcal{R}(n)$ the retrieved snippets retained for need n. Let $y(n, r)$ be the judge's label for snippet r. The number of supporting snippets is

$$s(n) = \sum_{r \in \mathcal{R}(n)} \mathbb{I}[y(n, r) = \texttt{support}], \tag{6}$$

and the percentage of needs with at least k supporting snippets is

$$\text{Coverage}\,@k = \frac{100}{|\mathcal{N}_{\text{eval}}|} \sum_{n \in \mathcal{N}_{\text{eval}}} \mathbb{I}[s(n) \geq k]. \tag{7}$$

Here, k is a support-count threshold, not a retrieval rank. The main comparison reports $k = 1$. A completed search with no retrieved results contributes zero support; `partial` labels do not count as support. Each need contributes equally, irrespective of how many snippets are retrieved.

Evaluation denominator. Coverage is computed over the evaluated sample, whereas # Needs reports the generated corpus size. In the saved TA evaluation, sampling up to four needs per analyst with seed 42 yields 632 records from 158 analysts, out of 1,333 generated needs. Of these, 516 have at least one supporting snippet, giving $100 \times 516/632 \approx 81.6\%$. The saved judgments contain 3,045 snippets and no recorded primary-judge errors. The computation does not interpret missing evaluations as completed negative judgments.

Semantic diversity. The diversity implementation encodes each need's description using a fixed `all-MiniLM-L6-v2` checkpoint.[2] Inputs are truncated to at most 256 tokens. Token representations are mean-pooled using the attention mask and normalized to unit length, yielding a 384-dimensional vector z_i for each need.

For a sampled set S of $m \geq 2$ need records, diversity is the mean cosine distance across all unordered pairs:

$$D(S) = \frac{2}{m(m-1)} \sum_{\substack{i<j \\ i,j \in S}} \left(1 - z_i^\top z_j\right). \tag{8}$$

Samples are drawn uniformly without replacement within each repetition, retaining duplicate records in the sampling pool. For R repetitions, the reported statistic is

$$\text{Diversity} = \frac{1}{R} \sum_{r=1}^{R} D(S_r). \tag{9}$$

Comparisons use the same sample size m across methods. The script exposes m as a run parameter and defaults to $R = 100$ repetitions with seed 42. Sampling uses all need descriptions in the relevant corpus, independently of whether external supporting evidence was retrieved.

Interpretation. Coverage@1 measures corroboration under the specified retrieval and judging procedure, rather than the prevalence of a need among practitioners. Diversity measures dispersion of need descriptions in the chosen embedding space; it does not by itself establish the validity or novelty of a task. The two metrics therefore characterize different properties of the elicited needs.

F REFERENCE TASK TAXONOMIES

We use two literature-informed reference taxonomies to assess which established analytical objectives are supported by the elicited needs. Their organizing principle is the *requested analytical objective and expected output*, rather than the analyst's occupation, application domain, or choice of algorithm. The TA taxonomy contains **12 task families**; the causal-inference taxonomy contains **13 task types nested within four families**. These are the respective denominators of Task Coverage in Table 4. All methods are evaluated against the same reference categories within each domain.

[2] Checkpoint revision: `1110a243fdf4706b3f48f1d95db1a4f5529b4d41`.

The tables below explain each category, with TA families numbered consecutively as T1–T12 for readability. Descriptions are condensed from the benchmark's source taxonomies; examples illustrate the categories and are not additional interview observations. These inventories serve as reference structures, not exhaustive accounts of possible analytical needs. Full coverage therefore means that each reference category has at least one supporting need, rather than that every subtype or possible workflow has been recovered.

F.1 Tabular Text Analysis: Task Families and Scope

The TA taxonomy distinguishes what users want to accomplish with text embedded in tables. This perspective accommodates both row-level operations and analyses that combine multiple rows or columns. Semantic operators such as filtering, joining, and aggregation provide an operational basis for several families (Patel et al., 2025); document-processing pipelines additionally illustrate extraction and summarization workflows (Shankar et al., 2025). Text clustering provides a separate setting in which the grouping itself must be discovered (Zhang et al., 2023). These works ground the included capabilities; the 12-family organization is the reference inventory used in this study.

Table 5: TA reference taxonomy. The 12 families are the units of Task Coverage; the right column makes their internal scope explicit.

ID and task family	Analytical objective → expected output	Included subtypes
T1 Classification & Labeling	Assign rows to specified categories or scales → a label or score for each row.	Sentiment; predefined topics; communicative intent; urgency or priority; compliance or risk labels.
T2 Semantic Grouping & Clustering	Discover categories from the text → groups and row memberships, potentially at multiple levels.	Topic discovery; grouping equivalent meanings; hierarchical grouping; grouping with outlier separation.
T3 Semantic Filtering	Retain rows satisfying a meaning-based condition → a subset of the input rows.	Semantic predicates; entity aliases; implicit conditions; negation or exceptions; cross-column conditions.
T4 Information Extraction	Recover information expressed in text → structured fields, relations, or event records.	Entities; numbers or dates; relations; open-schema attribute extraction; fixed-schema event extraction.
T5 Text-to-Table Structuring	Organize unstructured documents or complete table content → a structured table or filled cells.	Text-to-table conversion; semantic imputation from other rows; batch document ingestion.
T6 Text Aggregation & Summarization	Consolidate text across rows → group summaries, semantic counts, comparisons, or salient records.	Group summaries; topic counts; cross-group comparisons; temporal summaries; anomalous-row surfacing.
T7 Text × Numeric / Time Analysis	Analyze text jointly with numeric or temporal columns → conditional statistics, explanations, or trends.	Text-conditioned numeric aggregates; numeric filtering then text aggregation; text explanations of numeric anomalies; joint text/time trends.
T8 Cross-Text Semantic Matching	Align or compare multiple text fields → matched rows, joint labels, or inconsistency flags.	Multi-text fusion classification; cross-table semantic joins; cross-column contradiction detection.
T9 Semantic Ranking & Ordering	Order rows by a text-derived criterion → a ranked list, rather than only per-row labels.	Ordering by sentiment intensity, urgency, information density, or relevance to a query.
T10 Anomaly & Pattern Detection	Identify departures from semantic regularities → flagged outliers, duplicates, conflicts, or new signals.	Semantic outliers; near-duplicates; cross-column contradictions; emerging topics over time.

Continued on next page

Table 5 continued: TA reference taxonomy.

ID and task family	Analytical objective → expected output	Included subtypes
T11 Data Quality Repair	Standardize, correct, or complete text values → repaired cells or proposed corrections.	Spelling/format normalization; semantic imputation; anomalous-value flagging and correction.
T12 Semantic Augmentation & Column Generation	Transform or infer content from existing text → additional derived columns.	Per-row summaries; keywords or tags; translation; rewriting or canonicalization; inference-based derived columns.

Distinguishing related objectives. Known issue labels imply T1; discovering issue categories implies T2; returning only relevant rows implies T3. Urgency labels belong to T1, whereas an urgency-ordered list belongs to T9. T4 recovers stated information, including open-schema attributes; T12 also covers inferred fields and transformations. An unspecified output schema alone therefore does not establish an uncovered task. Per-row summaries are T12; summaries across rows are T6.

Overlap and coverage granularity. Families may overlap: semantic imputation appears in both T5/T11, and cross-column contradictions in T8/T10. The TA matcher assesses each family independently and permits multiple matches when each is supported by the requested objective and output. Subtypes are not separate coverage units. Purely numeric analysis, exact-key joins, and generic reporting do not establish a match without the corresponding text-semantic operation.

F.2 Causal Inference: Families, Task Types, and Boundaries

The causal inventory is informed by Causal-Copilot's coverage of discovery and inference workflows (Wang et al., 2025). Its four families distinguish requests for causal structure, intervention effects, counterfactual reasoning, and validity assessment. The 13 task types in Table 6 are the coverage units.

Table 6: Causal-inference reference taxonomy: four families and 13 task types, distinguished by their objectives and requested outputs.

ID and task type	Analytical objective → expected output	Illustrative request
A. Causal Discovery — identify causal structure		
A1 Pairwise Causal Direction	Assess the direction between two variables → a directional conclusion with supporting evidence.	Does demand drive price, or does price drive demand?
A2 Causal Graph / Structure Learning	Discover relationships among multiple variables → a causal graph or directed-edge table.	How do the production variables causally affect one another?
A3 Causal Feature Selection / Markov Blanket	Identify a target-relevant variable set → selected variables with causal or conditional-independence rationale.	Which columns form the outcome's Markov blanket?
A4 Temporal / Time-Series Causal Discovery	Discover directional relationships over time → a temporal structure with relevant lags.	Does earlier marketing spend drive later sales?
B. Causal Effect Estimation — quantify or act on intervention effects		
B1 Average Treatment Effect (ATE)	Estimate population-average impact → an overall effect estimate and uncertainty.	How much did the program improve outcomes on average?
B2 Heterogeneous / Subgroup Treatment Effect (CATE)	Estimate how impact varies → subgroup or covariate-conditional effect estimates.	Does the program help new customers more than existing ones?

Continued on next page

Table 6 continued: causal-inference reference taxonomy.

ID and task type	Analytical objective → expected output	Illustrative request
B3 Before/After / Temporal Causal Impact	Attribute a temporal change to an intervention → a time-aware effect relative to a counterfactual baseline.	Was the post-rollout drop caused by the policy?
B4 Mediation Analysis	Separate pathways through a mediator → direct and indirect effect estimates.	How much of training's effect operates through confidence?
B5 Policy / Targeting / Uplift	Decide where to intervene using incremental benefit → a targeting ranking or action recommendation.	Which customers should receive the offer?
C. Counterfactual Reasoning — explain alternative or observed outcomes		
C1 Counterfactual Estimation	Estimate an outcome under an alternative action → a counterfactual outcome and comparison with actuality.	What would revenue have been without the campaign?
C2 Anomaly Attribution / Root Cause Analysis	Explain an unexpected outcome causally → contributing factors or a causal decomposition.	Which factors caused the unexpected failure spike?
D. Causal Validity & Assumptions — assess the basis of a causal conclusion		
D1 Confounder Identification / Fair Comparison Check	Assess adjustment and comparability → candidate confounders, adjustment variables, or bias concerns.	What should we control for to compare treated and untreated groups?
D2 Sensitivity / Robustness Analysis	Assess dependence on assumptions → sensitivity estimates, robustness checks, or qualified conclusions.	Would the conclusion change under unmeasured confounding?

Category boundaries. Discovering which causal links exist (A) differs from estimating an intervention's impact (B). Within B, an average effect (B1), variation in effects (B2), and a decision about whom to treat (B5) request different results. B3 attributes a temporal change to an intervention; a raw before/after difference is insufficient. B4 decomposes an effect through an explicit mediator. C1 asks for an alternative outcome, whereas C2 asks which causes contributed to an observed anomaly. D1 concerns adjustment and comparability; D2 concerns the stability of a conclusion under changed assumptions. The causal matcher records a reference task type only when the request's objective and expected output support that assignment; shared causal terminology alone is insufficient.

Interpretation and limits. Data modality, estimator, sample size, and treatment type describe task instances rather than additional categories. The taxonomy describes requested analyses, not guarantees that the available data identify their answers. In particular, a Markov blanket is not synonymous with a set of direct causes (Pellet & Elisseeff, 2008), and a predictive lead–lag relationship alone does not establish a structural causal effect (Shojaie & Fox, 2021). Requests may require additional assumptions or data, or an explicit statement that the desired conclusion is not identifiable. Needs not adequately represented by the reference categories are retained for induction; the uncovered marker is not an additional reference task in the coverage denominator.