ScheMatiQ: From Research Question to Structured Data through Interactive Schema Discovery

Shahar Levy[1*] **Eliya Habba**[1*] **Reshef Mintz**[1]
Barak Raveh[1] **Renana Keydar**[2] **Gabriel Stanovsky**[1,3]
[1]School of Computer Science and Engineering, The Hebrew University of Jerusalem
[2]Faculty of Law, The Hebrew University of Jerusalem [3]Allen Institute for AI
{shahar.levy2, eliya.habba, gabriel.stanovsky}@mail.huji.ac.il
ScheMatiQ Website

Abstract

Many disciplines pose natural-language research questions over large document collections whose answers typically require structured evidence, traditionally obtained by manually designing an annotation schema and exhaustively labeling the corpus, a slow and error-prone process. We introduce ScheMatiQ, which leverages calls to a backbone LLM to take a question and a corpus to produce a schema and a grounded database, with a web interface that lets steer and revise the extraction. In collaboration with domain experts, we show that ScheMatiQ yields outputs that support real-world analysis in law and computational biology. We release ScheMatiQ as open source with a public web interface, and invite experts across disciplines to use it with their own data. All resources, including the website, source code, and demonstration video, are available at: www.ScheMatiQ-ai.com.

1 Introduction

Across disciplines, research often begins with a natural-language question posed over a large collection of documents. For example, consider real-world questions from different fields: a legal scholar asking, *Do judges appointed by different U.S. presidents differ in how they rule on immigration injunction cases?* in a large corpus of court decisions (Klerman, 2025); a computer scientist asking, *When is Chain-of-thought (CoT) really helpful?* across hundreds of NLP papers (Sprague et al., 2024); or a computational biologist investigating whether *It can be determined if a protein contains a nuclear export signal?* in a large collection of lab protocols (Xu et al., 2012).

Common to all such questions is the need to support answers with structured data over *observation units*, the primary elements of interest implied by the research question and the corpus (Blalock Jr, 1960). For example, in the legal domain, this may be a Supreme Court justice.

Obtaining structured data traditionally requires extensive manual effort across two mutually-informing stages. First, domain experts design an annotation schema that specifies the key question attributes (e.g., appointing president, ruling outcome) and potential confounders (e.g., age or education). Developing this schema requires domain knowledge and familiarity with the corpus. Second, annotators label the corpus according to the schema. This work, often delegated to research assistants, is expensive, slow, and vulnerable to human error (Artstein and Poesio, 2008).

Though such research efforts are very common, they are not well supported by current LLM-based technologies, including many "deep research" solutions. These systems are typically geared toward retrieval rather than exhaustive processing, and they produce outputs that are difficult to interact with, manipulate, or ground in the input texts.

In this work, we present ScheMatiQ, a framework that helps domain experts analyze large document collections around a guiding research question. As illustrated in Figure 1, ScheMatiQ leverages calls to a backbone LLM to identify observation units, induce an annotation schema, and generate a structured database, grounding each output in the source documents so users can verify the evidence behind it. A dedicated user interface lets experts iteratively steer the extraction process by inspecting and revising schema elements.

We evaluate ScheMatiQ on two real-world use cases, in close collaboration with domain experts in law and computational biology. These settings pose distinct challenges: legal analysis often hinges on long-form arguments, whereas computational biology frequently demands numerical, protocol-grounded reasoning. In both settings, ScheMatiQ generates structured outputs that matches the vast majority of human-annotated schemas and intro-

[*] Equal contribution.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations), pages 220–230
July 2-7, 2026 ©2026 Association for Computational Linguistics

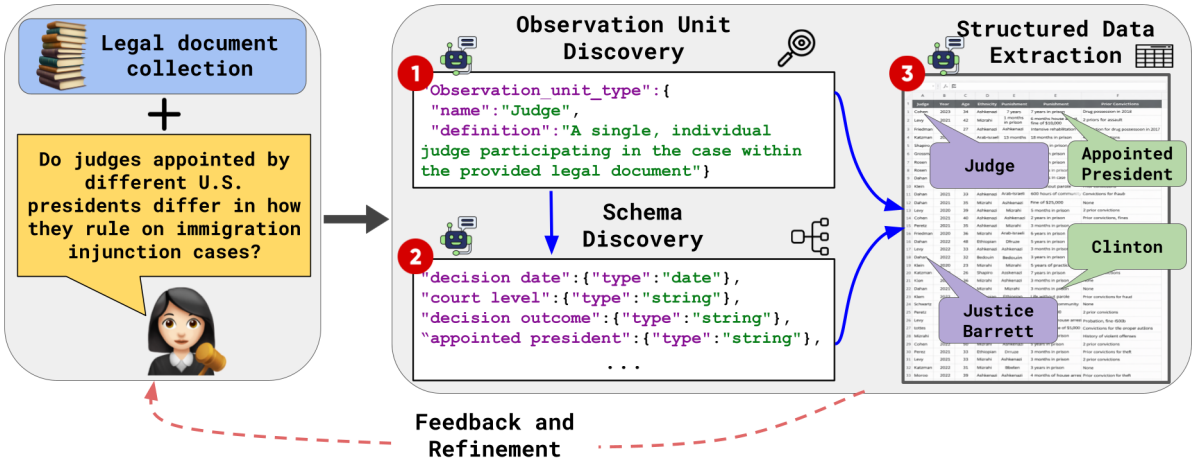


Figure 1: **ScheMatiQ workflow**. Given a natural-language question and a document collection, the system (1) discovers the observation unit, (2) discovers a query-guided schema, and (3) extracts structured values from the documents. Researchers can refine the schema and results through an interactive feedback loop.

duce new attributes that experts find useful.

We make ScheMatiQ fully open-source, and make it easy to use through a public web interface. We invite domain experts across disciplines to use it with their own questions and document collections, and NLP researchers to use it as a testbed for studying challenges such as long-context processing, efficiency, and effective user interfaces.

Our contributions are as follows: (1) We introduce ScheMatiQ, a framework for automatic schema discovery and structured data extraction from an expert's natural-language question and a collection of documents. (2) We design and implement an interactive web-based system that supports human–AI collaboration. (3) We conduct an evaluation with domain experts in two real-world domains, showing ScheMatiQ recovers human-annotated schemas while also adding new valuable information.

2 ScheMatiQ Principles

We design ScheMatiQ around three core principles that reflect the real needs of experts in various disciplines.

Query-Driven Discovery. ScheMatiQ grounds the entire pipeline in the *expert's natural-language query*. We will show that different research questions over the same documents can lead to different observation units and, in turn, different data structures.

Human-in-the-Loop. ScheMatiQ keeps experts in control by making every component editable. Since experts bring essential domain knowledge, the system is designed to integrate their feedback at every stage. This principle ensures that the final dataset reflects both the model's suggestions and the expert's expertise.

Grounded and Traceable Outputs. ScheMatiQ grounds each of its outputs in the source documents. This allows experts to verify results, assess extraction quality, trace unexpected outputs, and ultimately trust that the final dataset is reliable and interpretable.

3 ScheMatiQ

ScheMatiQ consists of three steps as illustrated in Figure 1. First, given a natural language query and a collection of documents, the system *discovers the observation unit*: the entity that each instance of the data should represent (Section 3.1). Second, using the documents, research question, and discovered observation unit, ScheMatiQ *discovers the schema* by iteratively refining the list of fields relevant to answering the question as it processes the documents (Section 3.2). Third, ScheMatiQ *extracts values* for the fields in the discovered schema across all documents, producing an output structured database (Section 3.3). Throughout the process, experts can revise both the schema and the extracted data through human–AI collaboration.

Below we elaborate on each of these steps, and provide prompt details in the Appendix C.

3.1 Observation Unit Discovery

The first step is to identify the observation unit type, defining the structure of the resulting data by specifying what object each instance represents (Blalock Jr, 1960).

For instance, in *Do judges appointed by different U.S. presidents differ in how they rule on immigration injunction cases?*, the type of the observation unit is a Supreme Court justice. For *When is Chain-of-thought helpful?*, the type is a single model evaluation under a specific experimental configuration. And for *Can it be determined whether a protein contains a nuclear export signal?*, the type is an individual protein.

The relationship between documents and observation units is many-to-many: a single document may discuss multiple observation units, and the same observation unit may be discussed in multiple documents. Figure 3 illustrates how different research questions imply different observation units and, in turn, different data structures and document–observation-unit relationships.

To identify the type of the observation unit, as illustrated in Figure 2a, we perform one LLM query using the expert's research question together with a batch of documents, asking it to "identify what type the query is asking for." The output of this step specifies the *observation-unit type*, along with a *description* of how it appears in the documents, and *example instances* either from the input documents or from the model's parametric data. These outputs are displayed in the web interface, as shown in Figure 4.

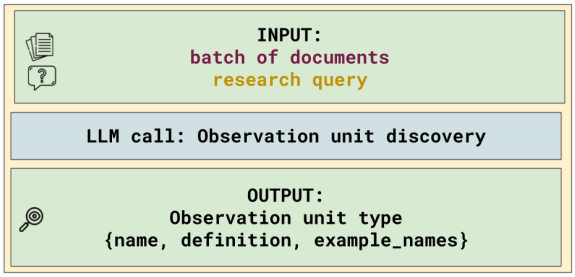
Human-in-the-Loop: Experts can revise the predicted type of observation unit or specify it manually if it is known in advance. This flexibility ensures that the resulting data will be structured around a desired entity.

3.2 Schema Discovery

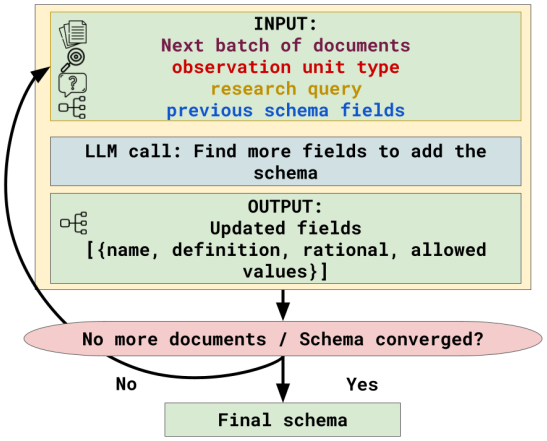
After identifying the observation unit type, we discover *the schema of the resulting data structure*: a set of attributes that describe each observation unit (e.g., a particular judge, experiment, or protein) in ways that are relevant to answering the research question. For example, the schema in Figure 1 includes, for each Supreme Court justice, the appointing president and the outcome of their decision, among other relevant fields.

Designing the schema is a crucial step in answering research questions over document collec-

(a) Diagram of the observation unit discovery flow.



(b) Diagram of the schema discovery flow.



(c) Diagram of the structured data extraction flow.

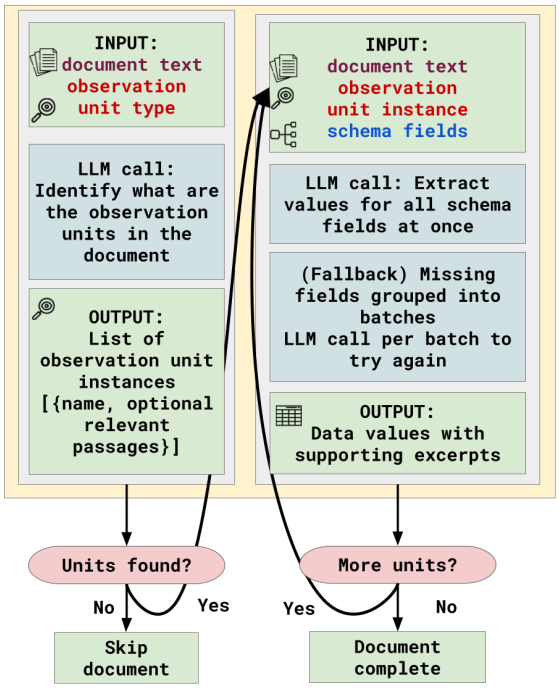


Figure 2: Diagrams illustrating the three system components described in Section 3. Each panel shows the corresponding stage in the pipeline.

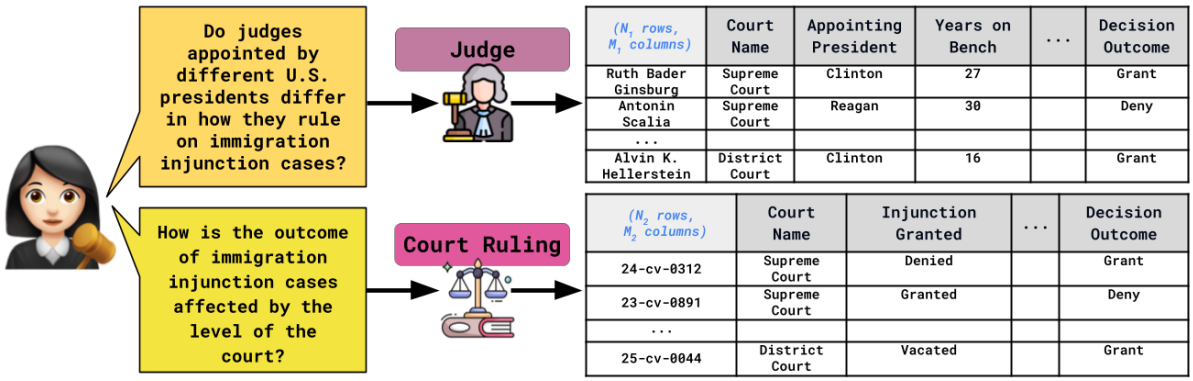


Figure 3: Different research questions over the same collection of documents lead to different observation units. A judge-level question (top) yields one row per judge, while a case-level question (bottom) yields one row per court ruling, resulting in different schemas and table structures.

tions, and it is traditionally constrained by human capacity. If key factors are omitted, the analysis may miss important explanations or confounders. For example, a judge's age or seniority could mediate decision-making, but would be invisible if not encoded in the schema. In manual workflows, schemas are typically shaped by the expert's domain knowledge, preconceptions, and familiarity with the corpus. These limitations become especially acute for large collections.

ScheMatiQ enables a more accurate, scalable human-computer workflow by leveraging LLMs to surface important attributes *across the entire document collection*. As illustrated in Figure 2b, we discover the schema by iteratively processing document batches and asking an LLM: "Do these documents suggest adding or refining the schema?". The output specifies, for each field, a free-form *definition* and a *rationale* explaining how the field supports answering the research question, along with optional *allowed values*, for example, whether the field should be numerical or free-form text. These outputs are displayed in the web interface, as shown in Figure 4, and support human verification. They are also consumed by the data-extraction step. This process repeats until no new fields are proposed or the corpus is exhausted.

Human-in-the-Loop: ScheMatiQ supports two forms of schema intervention: (1) Field editing: modifying definitions or adding, removing, and merging fields; and (2) Incremental discovery: adding new documents after initial convergence, prompting the system to propose additional fields while preserving the existing schema. These mechanisms enable iterative, flexible exploration as researchers expand their document collection and refine their understanding of the domain.

3.3 Structured Data Extraction

Once the observation unit and full schema are obtained, we use them to annotate the document collection. The resulting structured data is represented as a table whose rows correspond to observation-unit instances and whose columns correspond to the schema attributes. This step mitigates the need for laborious and error-prone human annotation, enables downstream analysis of the extracted data, and it allows researchers to assess the schema by observing the values which populate each column and whether they capture meaningful patterns across the corpus.

As illustrated in Figure 2c, extraction is done in two stages. For each document, an LLM first identifies all instances of the observation unit (e.g., "Ruth Bader Ginsburg", or "Antonin Scalia"). Then, for each instance, the LLM attempts to fill all schema fields in a single pass, and for any fields that remains unfilled, it performs a targeted follow-up extraction. All extraction is constrained by a strict evidence rule: a value can be extracted only if it is clearly supported by text in the document. Each output cell consists of the *extracted value* and the *supporting evidence* grounded in specific text from the input documents, and displayed for experts in the web-interface as shown in Figure 4.

Human-in-the-Loop: Experts may correct or refine extracted cells, ensuring that the structured data reflects accurate, evidence-supported values. They can also add additional documents, allowing the table to expand as new data becomes available.

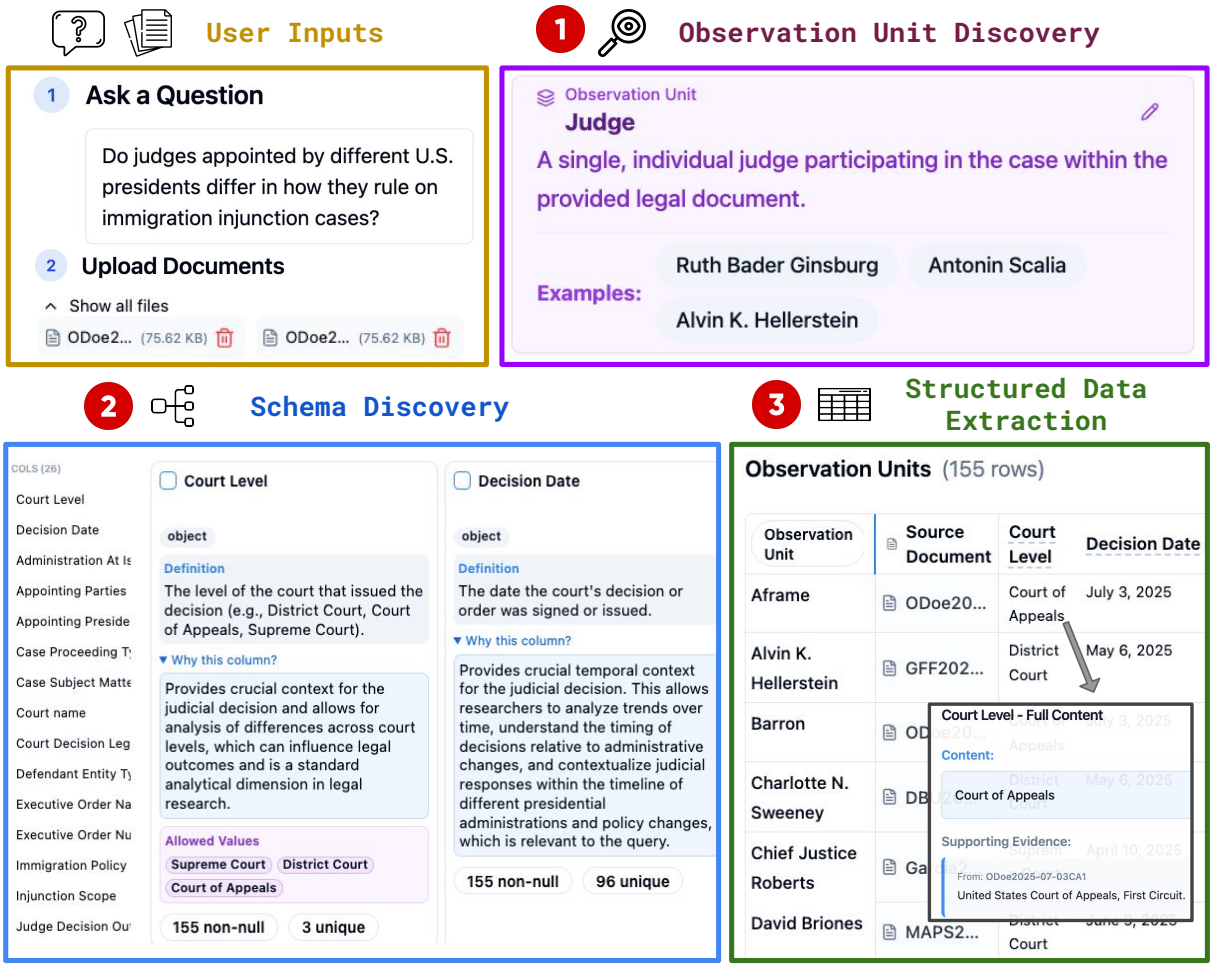


Figure 4: Screenshots of the ScheMatiQ web interface. Users provide a query and documents, inspect and refine the discovered observation unit and schema, and interact with the extracted table.

4 System Evaluation

Evaluating ScheMatiQ is challenging because it combines multiple components, human interaction, and large corpora of specialized texts, making direct end-to-end comparison to human annotation non-trivial. With domain experts, we study two use cases in empirical legal research and computational biology based on prior large-scale annotation projects, where a corpus, research question, schema, and human-annotated dataset already exist. This enables a direct comparison of ScheMatiQ's outputs to human annotations, measuring agreement, omissions, and novel fields.

While these benchmarks are extremely challenging, reflecting several person-years of expert effort, they should not be treated as pure gold standard. Human-annotated schemas reflect feasibility constraints and can contain human errors. Ultimately, the value of ScheMatiQ is best measured by its real-world impact (Reiter, 2025), i.e., its adoption for *new questions across disciplines*.

4.1 Experimental Setup

For each of our two domains, we specify the research question, the document corpus, and the human-annotated dataset. See additional implementation details in Appendix A.

In all experiments we use the Gemini-2.5 family (Comanici et al., 2025): Gemini-2.5-flash for observation-unit and schema discovery, and Gemini-2.5-flash-lite structured data extraction. The total cost of both of these uses cases is roughly 1 USD per 100 documents.

Users can specify other backbone LLMs by providing an API key to any model supported by `together.ai`.

Legal analysis. We follow Klerman (2025)'s analysis of 89 U.S. court decisions on immigration cases, asking *Do judges appointed by different U.S. presidents differ in how they rule on immigra-*

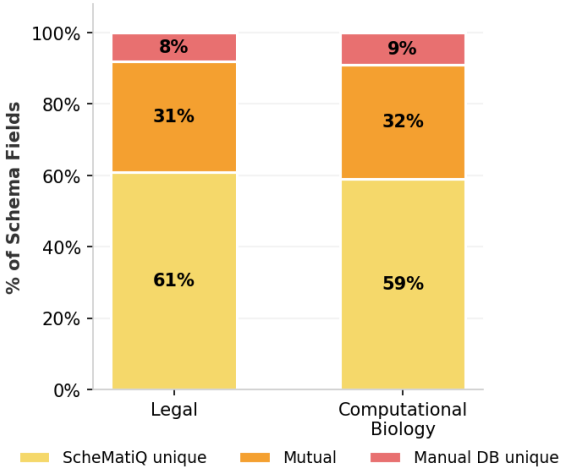


Figure 5: Schema-field coverage relative to manually curated gold schemas in the legal and computational biology domains. Bars show the proportion of fields unique to ScheMatiQ, shared with the manual DB schema, or unique to the manual DB schema.

tion injunction cases? To answer this, Klerman (2025) annotates each document with the judge name, appointing president, and decision outcome.

Computational Biology. We use NESdb (Xu et al., 2012), a manually curated dataset of protein annotations in 96 scientific articles, asking if *"it can be determined whether a protein contains a nuclear export signal? If so, how strong is it, and what is the confidence in that assessment?"*

4.2 Results

Below we outline interesting conclusions derived from our experiments using ScheMatiQ:

ScheMatiQ successfully recovers gold schemas and contributes new, relevant fields. Domain experts first align the manually curated schema with the schema discovered by ScheMatiQ, then evaluate the fields that are unique to each schema. Figure 5 shows the resulting distribution of manual-only, shared, and ScheMatiQ-only fields across the two domains. In both settings, ScheMatiQ recovers all but two broad miscellaneous fields. In contrast, the fields proposed by ScheMatiQ receive high relevance ratings, with mean scores of 4.2/5 in computational biology and 3.6/5 in the legal domain. For example, useful fields suggested in the legal domain include the legal basis for the court's decision, the scope of the injunction, and the presidential administration whose policy was challenged.

Domain	Attribute	Precision	Recall
Computational Biology	Export receptor	97.4%	90.7%
	Detection method	98.4%	75.9%
	Source organism	70.9%	64.8%
Legal	Court level	100.0%	100.0%
	Decision date	97.2%	97.0%
	Decision/Vote	99.2%	91.6%

Table 1: ScheMatiQ value-extraction precision and recall by domain.

ScheMatiQ's inputs are essential for capturing meaningful structure over real-world research questions. To assess the contribution of each input, we compare schemas generated under three configurations: using only the research question, using only the documents, and using both. Figure 6 shows that question-only schemas tend to produce high-level, generic fields (e.g., Judge Name, Protein ID), while document-only schemas introduce broad content that is not necessarily aligned with the research question. In contrast, combining both inputs yields richer, context-specific fields (e.g., Immigration Policy Context, Mutation Description). The absence of a three-way overlap indicates that meaningful schemas do not emerge from either input alone; real-world research questions require query-dependent schema discovery.

ScheMatiQ successfully recovers observation units, while there's room for improvement for documents with many observations. In computational biology, ScheMatiQ identifies 87% of proteins, and in the legal domain it identifies 97.5% of the judges in the human-annotated dataset, with 82% precision on the extracted set[1]. This highlighting its potential to automate expensive annotation. Error analysis in both domains shows that most misses occur in documents containing many observation units, while recall is near-perfect when documents mention a single entity. Future work can specifically target these high-density documents.

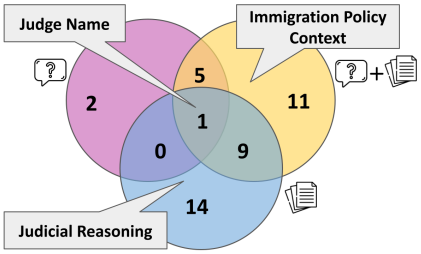
ScheMatiQ's value extraction is accurate but sensitive to normalization and missing evidence. In Table 1, we evaluate ScheMatiQ's value extraction, by first aligning the observation units it identi-

[1]The U.S. Supreme Court decides cases with all nine justices participating, and justices not named in an opinion are presumed to have joined the majority. The human-annotated dataset therefore records all nine justices for every Supreme Court decision. ScheMatiQ, by contrast, extracts only judges named in the document text. Accordingly, we do not count these unnamed justices as recall errors.

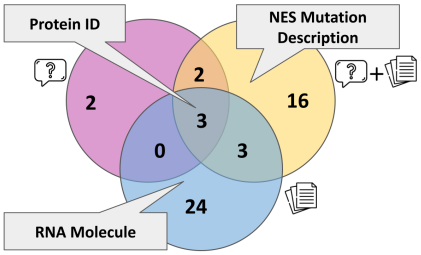
fied with the original dataset units and then measure cell-level precision and recall. In each domain, we focus on attributes which can be evaluated reliably.

In the computational biology domain, for the 162 proteins that were successfully aligned, two main patterns of errors emerge. First, *naming granularity*: many errors in attributes like *source organism* (60% of errors) and *export receptor* (13% of errors) come from extracting surface forms (e.g., "HPV-11," "Mouse") instead of the canonical names used in the human-annotated dataset (e.g., "Human papillomavirus type 11," "Mus musculus"). Second, the system shows consistently *high precision*, indicating that when ScheMatiQ extracts a value, it is usually correct. Lower recall is often because ScheMatiQ omits values when there is no explicit evidence in the text, rather than making incorrect extractions.

In the legal domain, for the 143 judges that were successfully aligned, ScheMatiQ reliably extracts attributes stated explicitly in the documents: *court level* with no errors and *decision date* with 97% accuracy. Most *decision date* disagreements are attributable to the human-annotated dataset, including a typographical error (year "3035" instead of "2025"). Errors concentrate on *decision/vote*, where the label requires interpreting a judge's alignment with the outcome rather than reading a surface form. In most cases, ScheMatiQ abstains rather than predicts incorrectly, lowering recall (92%) below precision (99%).



(a) Legal domain.



(b) Computational biology domain.

Figure 6: Schema-field overlap across three input conditions—query only (purple), documents only (blue), and the combined setting used by ScheMatiQ (yellow).

5 Related Work

Schema discovery from document collections has been studied across several settings. Early text-to-table methods learn document-to-table mappings from supervised pairs (Wu et al., 2022), without conditioning on a query. Later work generates literature-review tables whose rows are papers and whose columns capture aspects for comparison (Newman et al., 2024), separating schema generation from value extraction, optionally conditioned on a user-provided intent (Padmakumar et al., 2025). In all cases, rows remain paper-level. Other systems support query-driven extraction: Jiao et al. (2023) produce a tabular output from a single text without defining row-level structure across documents; SciDaSynth (Wang et al., 2025) generates tables from user questions, with columns driven by the questions rather than the corpus; and Schema-Miner (Sadruddin et al., 2025) mines schemas from a domain specification, targeting reusable ontologies rather than a specific research question. Dunn et al. (2022) fine-tune LLMs over scientific text for structured extraction but assume a predefined schema, and even with a fixed schema Ghosh et al. (2024) find ad-hoc scientific extraction unreliable without close expert review.

ScheMatiQ conditions schema discovery on *both* the research question and the documents, and explicitly identifies the *observation unit* that defines each row. Neither input alone yields context-specific schemas, and experts can revise the unit, schema, and extracted values at any stage, so ScheMatiQ supports specific research questions rather than generic document comparison.

6 Conclusion

We introduced ScheMatiQ, an interactive framework for query-driven schema discovery and dataset construction. Given a research question and a corpus, ScheMatiQ identifies the appropriate observation unit, induces a question-specific schema, and extracts a structured dataset that experts can iteratively refine. Across empirical legal research and computational biology, our evaluation shows that ScheMatiQ produces meaningful schemas and supports practical research workflows.

7 Limitations and Ethical Concerns

Our experiments rely on closed-source LLM APIs, which makes full reproducibility difficult to guarantee. We observe small variations between runs even with fixed parameters, likely due to non-deterministic decoding or unannounced model updates by the provider. While these differences are typically minor, they may lead to slight changes in column naming or value extraction across runs. ScheMatiQ supports using open-weight models hosted locally, which can mitigate this issue.

Data privacy. Users can opt in to have their data recorded for research purposes, otherwise, we do not store any session data.

8 Acknowledgments

This research was supported in part by Google.org and the Google Cloud Research Credits program for the Gemini Academic Program. We are grateful to Hadar Franco (Anicca.AI) for her valuable help with the web interface and deployment.

References

Ron Artstein and Massimo Poesio. 2008. Survey article: Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596.

Hubert M Blalock Jr. 1960. Social statistics.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, Luke Marris, Sam Petulla, Colin Gaffney, Asaf Aharoni, Nathan Lintz, Tiago Cardal Pais, Henrik Jacobsson, Idan Szpektor, Nan-Jiang Jiang, and 3416 others. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities.

Alexander Dunn, John Dagdelen, Nicholas Walker, Sanghoon Lee, Andrew S. Rosen, Gerbrand Ceder, Kristin Persson, and Anubhav Jain. 2022. Structured information extraction from complex scientific text with fine-tuned large language models. *Preprint*, arXiv:2212.05238.

Satanu Ghosh, Neal R. Brodnik, Carolina Frey, Collin Holgate, Tresa M. Pollock, Samantha Daly, and Samuel Carton. 2024. Toward reliable ad-hoc scientific information extraction: A case study on two materials datasets. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 15109–15123, Bangkok, Thailand. Association for Computational Linguistics.

Yizhu Jiao, Ming Zhong, Sha Li, Ruining Zhao, Siru Ouyang, Heng Ji, and Jiawei Han. 2023. Instruct and extract: Instruction tuning for on-demand information extraction. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10030–10051, Singapore. Association for Computational Linguistics.

Daniel M Klerman. 2025. Are trump judges different? evidence from immigration cases. *Evidence from Immigration Cases (September 15, 2025). USC CLASS Research Paper*, (2519).

Benjamin Newman, Yoonjoo Lee, Aakanksha Naik, Pao Siangliulue, Raymond Fok, Juho Kim, Daniel S Weld, Joseph Chee Chang, and Kyle Lo. 2024. ArxivDIGESTables: Synthesizing scientific literature into tables using language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 9612–9631, Miami, Florida, USA. Association for Computational Linguistics.

OpenAI. 2023. Gpt-4 technical report.

Vishakh Padmakumar, Joseph Chee Chang, Kyle Lo, Doug Downey, and Aakanksha Naik. 2025. Intent-aware schema generation and refinement for literature review tables. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 23450–23472, Suzhou, China. Association for Computational Linguistics.

Ehud Reiter. 2025. We should evaluate real-world impact. *Computational Linguistics*, 51(4):1419–1431.

Sameer Sadruddin, Jennifer D'Souza, Eleni Poupaki, Alex Watkins, Hamed Babaei Giglou, Anisa Rula, Bora Karasulu, Sören Auer, Adrie Mackus, and Erwin Kessels. 2025. Llms4schemadiscovery: A human-in-the-loop workflow for scientific schema mining with large language models.

Zayne Sprague, Fangcong Yin, Juan Diego Rodriguez, Dongwei Jiang, Manya Wadhwa, Prasann Singhal, Xinyu Zhao, Xi Ye, Kyle Mahowald, and Greg Durrett. 2024. To cot or not to cot? chain-of-thought helps mainly on math and symbolic reasoning. *ArXiv preprint*, abs/2409.12183.

Gemini Team. 2023. Gemini: A family of highly capable multimodal models.

Xingbo Wang, Samantha L. Huey, Rui Sheng, Saurabh Mehta, and Fei Wang. 2025. Scidasynth: Interactive structured data extraction from scientific literature with large language model. *Campbell Systematic Reviews*, 21(4).

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing.

In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Xueqing Wu, Jiacheng Zhang, and Hang Li. 2022. Text-to-table: A new way of information extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2518–2533, Dublin, Ireland. Association for Computational Linguistics.

Darui Xu, Nick V. Grishin, and Yuh Min Chook. 2012. Nesdb: a database of nes-containing crm1 cargoes. *Molecular Biology of the Cell*, 23(18):3673–3676.

A Use Cases: Full Specifications

Legal Domain Dataset. Court decisions of U.S. court cases concerning immigration policies and injunction proceedings.

Full Query. Do federal judges appointed by different Presidents (Trump vs. other Republican vs. Democratic) differ in their voting tendencies on immigration injunction cases? Do Trump-appointed judges tend to be more supportive of Trump administration immigration policies compared to judges appointed by other Republican or Democratic presidents?

Observation Unit — Judge. A single, individual judge participating in the case. If a case includes multiple judges (e.g., a panel), each judge is treated as a separate observation (row).

Full Schema (Columns). Judges On Panel; Appointing Presidents On Panel; Appointing Parties On Panel; Policy Instrument Purpose; Plaintiff Immigration Status Type; Policy Instrument Type; Policy Instrument Issuing Authority; Court Decision Legal Basis; Decision Date; Immigration Policy At Issue; Executive Order Name; Legal Challenge Grounds; Defendant Entity Types; Injunction Scope; Policy Instrument Date; Judge Names; Judge Decision Outcome; Case Subject Matter; Administration At Issue; Policy Instrument Target Group; Executive Order Number; Judge Decision Tendency; Court Level; Case Proceeding Type; Plaintiff Entity Types; Court Name.

Computational Biology Domain Dataset. A collection of 110 scientific papers describing experimental studies of Nuclear Export Signals (NES) in proteins. The papers correspond to references scraped from NESdb.

Full Query. Given a protein sequence, can it be determined whether or not it contains a nuclear export signal (NES)? If it does, how strong is the NES, and what is the confidence in that assessment?

Observation Unit — Protein. A single protein or polypeptide sequence evaluated for the presence, strength, or characteristics of a Nuclear Export Signal (NES).

Full Schema (Columns). NES Motif Count; Export Mechanism Type; NES Critical Residues; NES Presence Status; NES Activation Conditions; Regulatory Interacting Protein; NES Determination Evidence; NES Binding Affinity; NES Origin; NES Masking Agent; Competing Localization Signals; Export Receptor; NES Residue Coordinates; NES Identifier; NES Functional Impact; NES Transferability; NES Consensus Conformity; NES Strength Characterization; Protein Name; Reclassification Status; Source Organism; NES Conservation Status; Observed Subcellular Localization; NES Regulation Mechanism; NES Structural Domain; Identified NES Sequence.

B System Architecture

The architecture of ScheMatiQ is organized into three main layers:

- **Frontend:** A React application built with TypeScript and Tailwind CSS. It provides an interactive interface for configuring queries and uploading input documents, editing schemas, and exploring extracted tables, with real-time updates streamed from the backend.
- **Backend:** A FastAPI server that exposes REST endpoints for all pipeline operations. It also maintains a WebSocket channel to stream live progress updates (e.g., step-by-step extraction results) to the frontend.
- **Core Library:** A standalone Python package implementing the core ScheMatiQ components: observation-unit discovery, schema discovery, and value extraction. The library supports multiple LLM providers, including OpenAI's GPT-4 (OpenAI, 2023), Google's Gemini family (Team, 2023), and Together AI[2] models. For local deployments, it also supports open-weight models hosted through the HuggingFace Transformers library (Wolf et al., 2020).

This separation enables researchers to use the core algorithms programmatically through the ScheMatiQ Python package, while the web interface layers add session management, cloud storage (Supabase), and an interactive human-in-the-loop editing flow. The entire system is deployed on Railway using Docker containers for portability and scalability.

C Prompt Templates

In this section, we present the core prompt structures guiding the ScheMatiQ discovery pipeline in Figure 7.

[2] https://www.together.ai

Task

```
Given a query and sample document passages,
determine the appropriate **observation unit** – what each row in the extracted table should represent.
```

Instructions

```
The observation unit is the specific entity type the query asks about:
-Each document may contain ONE or MULTIPLE instances of the observation unit
-Your task is to identify WHAT specific entity type the query is asking about
-Even if a document discusses only one instance, consider WHAT that instance represents
**Critical Principle: One Row = One Answer to the Query**
```

Output Format

```
{{ "observation_unit": {{
    "name": "ShortName",
    "definition": "Full sentence describing what constitutes a single row in the table",
    "example_names": ["Instance1", "Instance2", "Instance3"]
  }},
  "reasoning": "Brief explanation of why this unit was chosen"}}
```

(a) Simplified prompt for observation unit discovery.

Task

```
You are building a schema to extract structured information from documents. Given passages from
documents, identify what types of information they contain that would help answer the query.
```

Instructions

```
**Step 1: Assess document relevance**
If passages lack information relevant to the query:
→ Return {{"document_helpful": false, "columns": []}}
**Step 2: If passages contain relevant extractable information**
- **If an existing schema is provided:**
  - Assume the schema is already COMPLETE
  - Only propose columns for genuinely MISSING information
- **If no existing schema is provided:**
  - Create ONLY the essential columns based on what information can be extracted
```

Output Format

```
Return valid JSON only:
{"document_helpful": true|false, "columns": [...]}}
```

(b) Simplified prompt for schema discovery.

Figure 7: Simplified LLM prompt excerpts illustrating two core stages of the system pipeline: (a) observation unit discovery and (b) schema discovery. Full implementation details and complete prompts are available in our GitHub repository.