Virtually every industry has a need to process vast amounts of unstructured data. From our journalist collaborators who have reviewed police reports at scale to find patterns of judicial bias [5], to our medical educator users who have transformed textbooks into flashcards [1], unstructured data tasks require complex reasoning. Large language models (LLMs) can do the reasoning these tasks require, but we lack the foundational principles for building systems that make LLM-powered data processing *reliable, cheap, and usable.*

**I build open-source AI-powered data systems**, tightly coupling techniques from databases and human-computer interaction. My approach to research is to study where the obvious ideas break down in large-scale real-world deployments, which can span multiple years. The resulting insights drive advances in system internals (highlighted in blue), and deepen our understanding of user interaction in data and AI systems (highlighted in orange). In my PhD, I designed and built the DocETL stack (docetl.org, 3.3K GitHub stars) for unstructured text analysis at scale [10, 11, 17]. DocETL has been deployed in applications across journalism, law, medicine, policy, finance, and urban planning. It was recently used by public defenders in California to analyze documents for two criminal trials, **marking one of the first uses of AI-powered data analysis in courtroom proceedings.** Our query optimization techniques are being integrated into industrial databases like Snowflake, and our ideas for AI evaluation and interface design have been adopted by LangChain [9], ChromaDB [7], and OpenAI [8]. My first-author work has appeared and won awards in top conferences: in databases (SIGMOD, VLDB), human-computer interaction (UIST, CSCW), and natural language processing (NAACL).

## Systems for Declarative LLM-Powered Data Processing

**Motivation.** DocETL is a declarative system for unstructured text analysis. I started building DocETL because journalists at Berkeley wanted to analyze **1.5 million pages** of public records, such as police reports [5]. Their goal was to extract and aggregate instances of police misconduct to build a public database—an effort that, as noted by the investigative journalism program's director [4], would take decades without LLMs! As a data systems researcher, I saw an opportunity to integrate LLMs into declarative data processing. Users could specify pipelines using familiar operators—`map`, `filter`, `reduce`—described in natural language, for LLMs to execute. For the misconduct extraction task, a user could write: `map("extract all instances of excessive force")` → `reduce("generate a summary for each type of excessive force incident")`. The `map` operator processes each document independently to extract instances, and the `reduce` operator aggregates extractions across documents to generate summaries grouped by incident type.

**Research.** Given a pipeline of data operators, each expressed in natural language, how does a system execute the pipeline? The simplest execution strategy compiles each operator into a single LLM prompt (per document). I implemented it and expected the main challenge at scale to be cost optimization, as in traditional databases. But accuracy was the bottleneck, even with powerful LLMs like Gemini-2.5-Pro. The issue wasn't that LLMs lack capabilities—they can extract, reason, and summarize remarkably well—but that users, who are not AI/ML experts nor should they need to be, were writing operators whose tasks were too complex for a single LLM call to handle. For example, an LLM can extract locations from documents, but how many it can *reliably* extract depends on document length and how many locations are present (i.e., query selectivity). Asking it to extract 50+ locations from a 200 page document in one call often leads to misses.

Drawing on principles from query optimization, where relational databases rewrite queries into equivalent but cheaper forms, I adapted the idea to LLM-powered data pipelines—using the idea of rewrites to *automatically* improve reliability. **My key insight was that we can systematically decompose complex operators into smaller, well-scoped LLM calls.** These decompositions fall into three broad categories: splitting input data, splitting logic in the task, or iteratively refining outputs using a "judge" LLM. I formalized these patterns as rewrite directives, which LLM agents instantiate into concrete operator configurations. For example, one directive rewrites a single `map` into a sequence of `split`, `map`, and `reduce` operators, allowing the system to process smaller chunks independently and aggregate results reliably. Enabling these rewrites required extending the operator set itself, to operators beyond relational algebra, such as `split` for document segmentation into chunks, `gather` for augmenting each chunk with surrounding context (e.g., section headers or document metadata), and `resolve` for reconciling inconsistent LLM-generated entities before grouping or aggregation. I then designed a top-down search algorithm—inspired by the Cascades framework for query optimization [6]—to explore these rewrites and select plans that maximize accuracy. While rewrite directives represent a significant departure from

the rigid, algebraic rewrite rules used in traditional database optimizers, the idea is already gaining traction: for example, Snowflake now uses a rewrite directive for join optimization [2].

Implementing these rewrite directives in DocETL's query optimizer improved recall by over 80% on the misconduct extraction task [10]. **I open-sourced the system, which now has over 3.3K GitHub stars, 29 contributors, and 500+ active Discord members.** Over the past year, I've presented this work at popular industry conferences [3], podcasts like TWIML and O'Reilly, and discussed the optimizer design with teams at Google, Snowflake, Redis, and ByteDance. Some teams, e.g., the Scottish Climate Intelligence Service, have even hired engineers specifically to author DocETL pipelines.

Once users got high-accuracy pipelines, they started expressing reservations about the cost of such pipelines. For example, the Scottish Climate Intelligence Service needed to process 100-page climate reports but found accurate pipelines prohibitively expensive, and Google researchers exploring rewrite directives for question-answering pipelines faced similar concerns. I collaborated with Google to extend rewrite directives for multi-objective optimization, enabling DocETL to generate a *Pareto frontier* of pipelines balancing cost and accuracy [17]. We introduced two new classes of rewrites: (1) "cheap" rewrites, such as changing the model, which adjust cost without additional LLM agent calls, and (2) code synthesis directives, which rewrite operators to perform deterministic or simple parts of a task (e.g., parsing, counting, deduplication) with code. These rewrites minimize the amount of work delegated to language models, often reducing both input size and reasoning "depth" per call. Across six workloads, the new optimizer achieved up to 2× higher accuracy than prior systems' best pipelines and matched their accuracy at 37% of the cost. On a real criminal-defense workload from public defenders in a major California city, the optimizer found a pipeline that delivered **1.5× higher F1** than an expert-written baseline pipeline that used GPT-5, at only **5% of the cost**.

Overall, DocETL has sparked follow-up efforts from many students in my advisor's group, e.g., rewrite directives with probabilistic guarantees on accuracy relative to the original pipeline [15, 18]. Through DocETL's adoption and my outreach, **I independently secured over $100K in direct research funding and $25K in compute credits**. I also established many more collaborations across high-impact domains like medicine (with UCSF researchers) and sustainability (with California Water Consortium) for multi-month deployments.

### Interfaces for Steering LLM-Powered Data Processing Pipelines

**Motivation.** When biomedical researchers began using DocETL to extract insights from research papers, they emailed me, expressing the difficulty of authoring good pipelines. It was not for lack of domain knowledge—they knew exactly what insights they wanted—but they struggled to express those goals in complex, ambiguous natural-language prompts. At first, I created a simple chat interface with a graphical pipeline builder. Users could ask a question, and the system would automatically create a pipeline or adjust operators in response. But the chat-based approach frustrated even expert users: they did not know what to ask, how to ask it, or when something had gone wrong.

**Research.** Looking beneath these usability issues, I realized they reflected deeper *semantic gaps* between users, their pipelines, and the data those pipelines operated on. **My key insight was that user-facing challenges in AI-powered data analysis could be systematically framed as three *gulfs***, shown in Figure 1: *comprehension* (seeing and interpreting data and results), *specification* (expressing intent and constraints clearly), and *generalization* (coping with the inherent unreliability of LLMs at scale). The first two gulfs represent long-standing issues in data analysis, but the third—generalization—**is unique to LLM-powered workflows.** Because LLMs are stochastic and fail to generalize predictably, pipelines that appear correct on a few examples often degrade on others in subtle ways. In DocWrangler [11], we built features to bridge the three gulfs—for example, to bridge the gulf of specification, a mode that encourages reviewing DocETL outputs individually, taking notes, and automatically turning those notes into new operations. DocWrangler's online deployment has powered **over 3,500 pipelines created over the last 6 months**, with users running their analyses through their own paid API accounts, across domains such as environmental

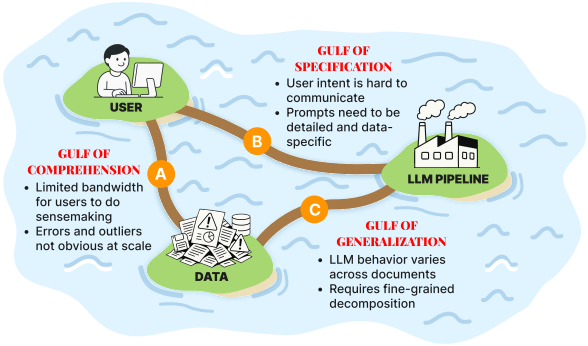


**Figure 1** "Three Gulfs" of LLM-Powered Data Processing.

analysis (US EPA), customer support (AT&T), and government reporting (Singapore's Ministry of Trade and Industry). This work received a **Best Paper Honorable Mention** at UIST 2025.

From a systems perspective, the three gulfs framework enables precise diagnosis: when users struggle, we can identify which gulf is the bottleneck and evaluate solutions systematically rather than addressing symptoms in an ad-hoc manner. From an HCI perspective, the broader impact lies in the methodology that led to discovering the three gulfs—deriving insights not from small, controlled studies, but from observing thousands of real pipelines across domains in a broad "lateral" deployment. AI systems uniquely generate a long tail of interactions, and recognizing patterns across that diversity may point to a new empirical foundation for studying how people build, adapt, and reason about AI systems at scale. Beyond DocWrangler, I use the three gulfs as the conceptual foundation for a course on AI engineering and evaluation, taught to **over 3,000 industry practitioners** in the summer of 2025. Participants described the framework as *"the biggest thing I took away from the course...I used to tackle all three at once, and just kept spinning [my] wheels"* (ML Engineer at Home Depot). A staff software engineer at a major online therapy platform said, *"This model completely reframed how I approach understanding and improving LLM applications...I plan to apply it to every new feature I work on."* The three gulfs have moved beyond research, becoming a shared language for reasoning about how LLM systems break—and how to fix them.

### Automatic Evaluation of LLM Systems

When optimizing pipelines in DocETL, we needed a way to estimate accuracy automatically. Rewrites can produce hundreds of new pipeline variants. Fine-tuning smaller LLMs to act as evaluators seemed appealing, but most LLM-powered tasks—e.g., extraction, reasoning, summarization, report generation—require difficult judgments that could take months of human labeling to reproduce. I wrote two papers on this problem, one at VLDB 2024 [12] and one at UIST 2024 [14], showing how accuracy in LLM-powered data processing can be estimated without labeled data. **My key insight was that signals for evaluation already exist in how users iterate on their pipelines.** When users edit prompts, emphasize specific ideas, or rephrase instructions, those changes implicitly express what was wrong and what "better" means. We developed lightweight methods to automatically mine pipeline edit histories, infer binary evaluation criteria (e.g., "answers should cite evidence," "output must stay under 200 words"), and synthesize code-powered or LLM-as-judge evaluators. An integer linear program then selects a minimal subset of these evaluators that collectively explain most observed LLM errors. We also built a lightweight interface for grading outputs, which surfaces these inferred criteria to users and refines them in the background as they provide feedback [14]. Through a user study, we surprisingly found that users' notions of correctness evolved as they graded outputs—a phenomenon we dubbed criteria drift. As users encountered new model behaviors at scale, they updated their own evaluation standards, revealing that even "objective" assessment is dynamic and negotiated over time. To support further research, I also open-sourced the *PromptEvals* dataset, which captures over 2,000 real-world pipeline edits and evaluation criteria [16]. Overall, though first developed for batch LLM data processing, my work has contributed foundational ideas for LLM evaluation more broadly and has already had measurable impact: **my UIST 2024 paper [14] is the most cited paper of the venue that year, and the underlying ideas now underpin tools developed by LangChain [9, 13], ChromaDB [7], and OpenAI [8].**

### Future Work

The DocETL stack's open-source nature and real-world use across journalism, law, medicine, and policy make it an ideal testbed: I can rapidly prototype new techniques, deploy them to hundreds of active users, and study how my ideas perform on diverse tasks. Here, I outline three major directions for my future work: a *deep* systems agenda that pushes the technical limits of unstructured data processing, a *broad* expansion of tools that support the full lifecycle of unstructured analysis, and a *"moonshot"* investigation into how AI changes knowledge work across many different domains.

**"Full-Stack" Unstructured Data Systems.** Almost all LLM-powered data systems, including DocETL, depend on commercial LLM APIs. But this is increasingly untenable. At scale, renting GPUs to run open-source models locally is often cheaper—API costs grow linearly, while self-hosting benefits from batching and amortizing GPU time. Also, many users cannot use commercial APIs at all; e.g., UCSF researchers need HIPAA compliance for patient notes. *How can we design data systems around open-source LLMs?* We need to rethink both inference and optimization. At the inference layer, document-processing workloads exhibit heavy reuse that existing engines ignore. Unlike chat applications, all inputs are

known in advance, operations are homogeneous, and execution order is flexible. I am interested in building LLM caching and unstructured data compression techniques that exploit this structure. Then, at the optimizer layer, DocETL can take hours due to two bottlenecks: calling an external LLM agent for every rewrite proposal, and executing candidate pipelines on samples to estimate costs and accuracies. But we now have traces from thousands of runs—enough to train agents that apply rewrite directives directly, and perhaps even predict accuracy gains from rewrites *without* executing pipelines on samples. Ultimately, open-source LLMs are now capable of high-quality analysis; we should design systems around them.

**Supporting the Full Lifecycle of Unstructured Data Analysis.** Unstructured data analysis lacks cohesive support across its full lifecycle—from identifying what to analyze, to authoring and refining pipelines, to communicating results. DocWrangler and DocETL fill crucial gaps in pipeline authoring, optimization, and execution, but users still struggle at both ends of the lifecycle. Before authoring, analysts need ways to explore large collections of documents to decide *what* questions to ask. If collaborating on analyses, teams need shared environments to iteratively refine pipelines and manage provenance as prompts, models, and evaluation criteria evolve together. After analysis, they need interfaces to help communicate findings that are often interpretive and non-deterministic. Building these tools will require new methods of visualization, communication, and uncertainty quantification, all tightly coupled with systems support.

**Understanding How AI Capabilities Reshape Society.** In DocETL deployments, I've observed that once people begin using AI to analyze data, their goals and workflows evolve in unexpected ways. Our public defender collaborators, for example, initially used DocETL to extract racial epithets from court records, but after seeing results across cases, they realized several court records were missing. They then repurposed their DocETL pipelines to identify the missing data. I believe adopting AI tools does not simply automate existing workflows; it transforms them in ways we cannot expect, because users must continually adapt their practices as model capabilities expose new possibilities and limitations. I want to study how such workflows evolve and stabilize across domains, and to develop conceptual models that explain—and eventually help anticipate—how AI adoption reshapes knowledge work.

Another way DocETL changes data work is by expanding the *kinds of data* that can be collected and studied. For instance, our collaborators at the San Francisco Bar Association are now laboriously collecting charging documents (i.e., records listing the charges in each case), so DocETL can extract structured features from them, enabling statistical analyses to inform policy. More broadly, the ability to analyze unstructured data at scale allows researchers to design new kinds of studies—for example, social scientists can analyze thousands of hours of interviews, and policymakers can synthesize citizen feedback en masse. I want to build systems that help domain experts develop *new methodologies* for studying data.

**Summary.** Overall, the directions I have outlined demand expertise across many areas of computer science—e.g., computer systems design for accelerated and low-cost inference; ML and NLP for learned query rewrites; HCI methods to study user needs and develop design principles and theories of technology-assisted work. I am excited to draw on my interdisciplinary skills to advance this vision, to collaborate with students and faculty across areas, and to establish new foundations for how humans and machines create, organize, and make sense of the world's unstructured information.

## References


[1] The anking step deck. Website, AnkiHub, 2024. Available via AnkiHub: https://www.ankihub.net/step-deck.

[2] Paritosh Aggarwal, Bowei Chen, Anupam Datta, Benjamin Han, Boxin Jiang, Nitish Jindal, Zihan Li, Aaron Lin, Pawel Liskowski, Jay Tayade, Dimitris Tsirogiannis, Nathan Wiegand, and Weicheng Zhao. Cortex aisql: A production sql engine for unstructured data, 2025.

[3] AI News. Docetl: Agentic query rewriting and evaluation for complex document processing. Newsletter post on Buttondown.

[4] David Barstow. David barstow talks berkeley's experiment to accelerate the production of investigative reporters. https://vcresearch.berkeley.edu/news/david-barstow-talks-berkeleys-experiment-accelerate-production-investigative-reporters, April 2025. Accessed: 2025-10-22.

[5] Berkeley Institute for Data Science (BIDS). California police records access project. https://bids.berkeley.edu/california-police-records-access-project, 2025. Accessed: 2025-10-05.

[6] Goetz Graefe. The cascades framework for query optimization. *IEEE Data(base) Engineering Bulletin*, 18:19–29, 1995.

[7] Kelly Hong, Anton Troynikov, Jeff Huber, and Morgan McGuire. Generative benchmarking. Technical report, Chroma, April 2025. Chroma technical report, April 7, 2025.

[8] Neel Kapse and Hamel Husain. Building resilient prompts using an evaluation flywheel. https://cookbook.openai.com/examples/evaluation/building_resilient_prompts_using_an_evaluation_flywheel, October 2025. Accessed: 2025-10-22.

[9] LangChain. Aligning llm-as-a-judge with human preferences, June 2024. LangChain blog, 5 min read.

[10] **Shreya Shankar**, Tristan Chambers, Tarak Shah, Aditya G Parameswaran, and Eugene Wu. Docetl: Agentic query rewriting and evaluation for complex document processing. *Proceedings of the VLDB Endowment*, 18(9):3035–3048, 2025.

[11] **Shreya Shankar***, Bhavya Chopra*, Mawil Hasan, Stephen Lee, Björn Hartmann, Joseph Hellerstein, Aditya Parameswaran, and Eugene Wu. Steering semantic data processing with docwrangler. In *Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology*, UIST '25, New York, NY, USA, 2025. Association for Computing Machinery.

[12] **Shreya Shankar**, Haotian Li, Parth Asawa, Madelon Hulsebos, Yiming Lin, JD Zamfirescu-Pereira, Harrison Chase, Will Fu-Hinthorn, Aditya G Parameswaran, and Eugene Wu. spade: Synthesizing data quality assertions for large language model pipelines. *Proceedings of the VLDB Endowment*, 17(12):4173–4186, 2024.

[13] **Shreya Shankar**, Haotian Li, Will Fu-Hinthorn, Harrison Chase, J.D. Zamfirescu-Pereira, Yiming Lin, Sam Noyes, Eugene Wu, and Aditya Parameswaran. Spade: Automatically digging up evals based on prompt refinements, November 2023. LangChain Blog, "6 min read".

[14] **Shreya Shankar**, JD Zamfirescu-Pereira, Björn Hartmann, Aditya Parameswaran, and Ian Arawjo. Who validates the validators? aligning llm-assisted evaluation of llm outputs with human preferences. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*, pages 1–14, 2024.

[15] **Shreya Shankar**, Sepanta Zeighami, and Aditya Parameswaran. Task cascades for efficient unstructured data processing. *Proceedings of the ACM on Management of Data (SIGMOD)*, 2026.

[16] Reya Vir*, **Shreya Shankar***, Harrison Chase, Will Fu-Hinthorn, and Aditya G. Parameswaran. Promptevals: A dataset of assertions and guardrails for custom production large language model pipelines. In *North American Chapter of the Association for Computational Linguistics*, 2025.

[17] Lindsey Wei*, **Shreya Shankar***, Sepanta Zeighami, Yeounoh Chung, Fatma Ozcan, and Aditya G. Parameswaran. Multi-objective agentic rewrites for unstructured data processing. In Progress, 2025.

[18] Sepanta Zeighami, **Shreya Shankar**, and Aditya Parameswaran. Cut costs, not accuracy: Llm-powered data processing with guarantees. *Proceedings of the ACM on Management of Data (SIGMOD)*, 2026.