Data Discovery is a Socio-Technical Problem: the Path from Document Identification and Retrieval to Data Ecology

Raul Castro Fernandez
Department of Computer Science, The University of Chicago
raulcf@uchicago.edu

Abstract

Data discovery is the task of identifying and retrieving documents that satisfy an information need—a need that may support decision-making, scientific inquiry, or operational insight. While the problem has existed since data began to be recorded, its complexity has grown with the scale and diversity of data, particularly tabular data in databases, spreadsheets, and cloud stores. Unlike well-defined tasks in data management, data discovery is ambiguous and socio-technical in nature, requiring both algorithmic and human-centered solutions. In this perspective, we conceptualize data discovery as distinct from but related to data search, introduce a reference architecture that separates identification from retrieval, and situate recent developments—including LLMs and retrieval-augmented generation—within this framework. We argue that solving data discovery demands embracing its socio-technical character and outline an emerging research direction, data ecology, that addresses this broader challenge.

1 Introduction

Data discovery is the process of identifying and retrieving documents that satisfy a specific information need. An *information need* refers to any data that supports a task—for example, a paragraph that answers a question, a row in a spreadsheet that informs a decision, or a feature that improves a predictor's performance beyond a given threshold. The task of *identifying* involves articulating the information need in a way that makes it possible to find relevant data. The task of *retrieval* involves gathering documents that fulfill this need. Defined broadly, the data discovery problem has existed since humans first began representing knowledge in the form of documents. Library science—concerned with how to organize and retrieve documents based on information needs—emerged after the printing press made books widely available. Later, database and information systems developed rapidly as organizations accumulated large document collections in the postwar era. Information retrieval and web search arose in response to the explosion of online (web) documents. Today, prompt engineering serves as a heuristic for retrieving relevant outputs from large language models (LLMs), given an information need expressed as a prompt. In this way, data discovery has evolved in step with both the scale of the problem and the technologies available to address it.

Data discovery is a critical problem in the context of tabular data. A significant portion of high-value data[1] is stored in tabular formats across databases, spreadsheets, and cloud storage systems. Data discovery for tabular data has been explored by the data management community and, more recently,

[1]We use high-value data informally to refer to documents that directly relate to an organization's operations, including those that contribute to value creation such as revenue, user engagement, and similar metrics.

by the machine learning community. It is also closely connected to fields like information retrieval and human-computer interaction (HCI), reflecting the diverse challenges involved in tackling this complex problem. Data discovery over tabular data has numerous applications. It supports scientific advancement by identifying documents that either validate a hypothesis or help formulate new ones (hypothesis-driven discovery). It also assists organizations in data-intensive tasks such as decision-making, product and service development, and more.

Some problems in data management resist precise definitions and are inherently ambiguous. This class includes tasks such as data cleaning, data wrangling, and data integration; data discovery belongs to this group too. When SQL is viewed as a language for retrieving documents that satisfy an information need—as specified by a query—the results are typically unambiguous, thanks to the closed-world assumption that enables clear and well-scoped data processing. However, once we acknowledge that relevant documents may span multiple databases and repositories—as is often the case in practice—the problem becomes more difficult to define and, consequently, more challenging to solve.

In this perspective article, we conceptualize the problem of data discovery, situating it in relation to data search and related tasks, and explain how it contributes to unlocking value from data (Section 2). We then present a landscape of solutions for data discovery over tabular data (Section 3), structured around a reference architecture that highlights the distinction between identification and retrieval. This architecture also serves as a framework to introduce key contributions from our group. Building on this foundation, we examine the emerging role of large language models (LLMs) and the evolving architectures for retrieval-augmented generation (RAG) and agentic systems in shaping the future of data discovery (Section 4). This discussion sets the stage for one of the paper's central claims: data discovery is fundamentally a socio-technical challenge. As such, no purely technical solution—not even those based on LLMs—can fully address it. Instead, we advocate for socio-technical approaches and introduce an emerging line of research, data ecology, that explores this broader framing (Section 5). We conclude with a discussion of the current state of the problem and outline what is needed to advance future solutions.

2 Conceptualizing Data Discovery

In this section, we introduce a set of definitions to support the discussions that follow (Section 2.1) and examine the role of data discovery in enabling value extraction from data (Section 2.2).

2.1 Basic Definitions

Data and Documents. We draw a clear distinction between data and documents [6]. Data refers to an abstract representation of some aspect or condition of reality, while a document is a representation of data. The same data may be represented in different documents. For instance, consider the concept of "room temperature": the data corresponds to the measured value itself, whereas a document might be a handwritten note recording the temperature, a value in a CSV file, or a row in a database table.

Documents exist in many forms. This heterogeneity is a core challenge of data integration and by extension data discovery.

Definition 2.1 (Information Need) *The data necessary to address a task.*

We refer to it as an information need rather than a data need because the ideal data is not always directly accessible; instead, we often rely on data that indirectly reveals information about the target. For example, if the goal is to predict whether a loan application should be approved, the ideal data would be whether the customer will default. However, that data may not be directly available. Instead, we might have access to related data—such as credit history or financial status—that provides indirect evidence about the likelihood of default. In such cases, an analyst may articulate their information need as the set of attributes (data) that relate to a customer's probability of default.

A key aspect of information needs is that they are articulated by an agent—human or bot—as part of pursuing a task. For example, the scenario illustrated in Figure 1 depicts multiple agents within an organization, each facing a data discovery problem. The information need expressed by the decision maker (bubble 1) may be high-level—for instance, "we need to project sales for this new product for next quarter." This initial request triggers several subtasks, such as reporting on past sales, assembling features for a predictive model, and ultimately compiling these derived data products into a report to support decision-making. The key point is that, when expressed by humans, information needs are typically formulated so that the next person in the workflow can interpret and act on them. Regardless of how an information need is expressed, the central challenge of data discovery is to identify which documents satisfy it.

Definition 2.2 (Data Discovery) *The problem of identifying and retrieving documents that satisfy an information need.*

In data discovery, the solution is a document. In some cases, this document already exists, and the task is simply to locate and retrieve it. This is typical of web search, information retrieval, or dataset search, where the goal is to identify existing webpages, documents, or tables. In other cases, however, the document does not exist in a ready-made form and must be constructed—through integration or fusion—as part of the discovery process. For example, multiple documents may need to be combined and transformed to produce a new document that satisfies the user's information need. Similarly, large language models (LLMs) combined with search can synthesize a document from information scattered across multiple sources, as seen in systems like Google's Deep Dive Search, Perplexity.ai, and ChatGPT with retrieval.

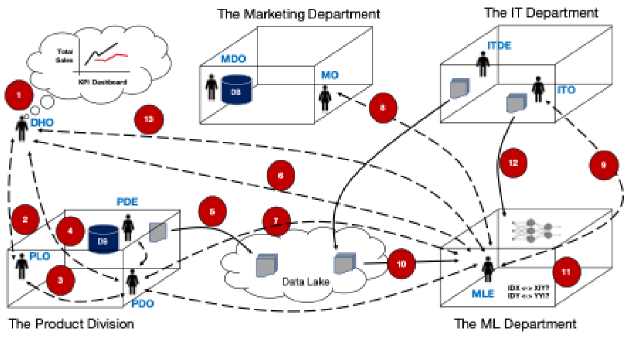

Figure 1: Simplified diagram of an organization where multiple stakeholders with information needs (indicated by the red numbered circles) interact with data systems and among themselves to solve a data task

Identify and Retrieve. Two key terms in the definition of data discovery are *identify* and *retrieve*. *Retrieving* refers to the mechanistic process of delivering a document to the user so they can proceed with their task. *Identifying*, by contrast, involves determining which document satisfies an information need—that is, which document represents data that matches the articulated need. *Identifying* and *retrieving* are related to the separation between cognitive-viewpoint and document-based view of the "anomalous states of knowledge" theory of information retrieval [3]. Successfully solving this process entails several challenges:

1. it requires a precise and unambiguous articulation of the information need, which in turn depends on the user's understanding of the task they are trying to accomplish;
2. the information need must be expressed in a form that the discovery system can use to match against available data—for example, by explaining a request to a librarian, entering keywords into a search engine, or writing a natural language query to an LLM;
3. the discovery system must be capable of interpreting the information need and searching across large volumes of data to find relevant matches.

We distinguish between *content* and *context* information needs [2]. Content information needs can be addressed by examining the contents of documents themselves. However, identifying a relevant document is not always possible through content alone. Context information needs require metadata about the document—such as who created it, how it has been processed, and other provenance—related attributes.

A critical piece of metadata in data discovery is a document's *provenance*. Provenance is essential for verifying the origin of a document and the chain of agents who created or modified it. In a world flooded with data—much of it of questionable quality—provenance becomes increasingly valuable, often serving as the primary means of distinguishing reliable information from noise. We emphasize provenance because, in manual discovery processes, it is typically maintained informally by the people involved, who can explain how a document was produced if asked. As discovery processes become more automated, however, preserving provenance requires deliberate effort to capture the **tribal knowledge** that resides in the minds of those humans. A central goal of data discovery is to encode this implicit knowledge into documents so they can be used more effectively to satisfy information needs—a topic we expand on later in the paper.

2.2 The Role of Data Discovery in the Data Value Chain

To understand the value of data discovery solutions, it is useful to situate them within the broader process of deriving value from data. Data has instrumental value—that is, it generates value when used to perform a specific task. However, data is not accessed directly; it is accessed through documents. The utility of a document for a given task depends both on the value of the data it represents and the cost of using that document [6]. We conceptualize the process of deriving value from data as a sequence of the following steps:

1. Formulate the information need associated with the task.
2. Discover a document that satisfies the information need. This document may already exist within the organization or may need to be assembled from multiple existing documents.
3. Analyze and process the document to produce a solution to the data task.

Data discovery plays a central role in the second step of the process, and—as we will argue later—it must also support the first step by helping agents articulate their information needs. The cost of data discovery reflects the time, effort, and resources required to identify documents that satisfy an information need. Because this cost is nonzero, it reduces the overall utility of the data. Therefore, the purpose of data discovery solutions is to minimize these costs, thereby increasing the utility that can be derived from data.

Most data management technologies function as instruments for reducing costs, and data discovery fits this role too. But data discovery is more than a cost reduction technique.

Beyond Cost Reduction. In the conceptual model outlined above, the first step is to formulate an information need arising from a data task. However, selecting which data task to pursue—among many possibilities—is itself a critical decision that shapes the value derived from data. Data discovery can play a generative role in this process by helping to surface new questions (as in hypothesis-driven discovery) that lead to the formulation of new tasks. At its best, data discovery does more than reduce costs: it can help define entirely new use cases, opening up fresh avenues for value creation.

3 Landscape of Data Discovery Solutions for Tabular Data

We present an overview of solutions for tabular data discovery. Our goal is not to provide a comprehensive survey of the literature[2], as such a review would exceed the scope of this paper. Instead, we focus on highlighting key ideas proposed in this area and clarifying how they relate to one another. This technical foundation will serve as the scaffolding for our central argument in the next section: that data discovery is a socio-technical problem.

The literature on data discovery can be categorized based on the emphasis and objectives of each contribution. Some work focuses on identification interfaces, while others emphasize retrieval mechanisms. A substantial portion of the research centers on foundational building blocks—techniques designed to support navigation through large repositories of tables. In Section 3.1, we introduce a reference architecture that we have recently used to illustrate how the challenges of tabular data discovery are interconnected. We then provide an overview of our group's work in this area and relate it to other relevant efforts to contextualize our main contributions (Section 3.2).

3.1 Applying Separation of Concerns to Data Discovery

A reference architecture embodies a *separation of concerns* strategy, which breaks down complex engineering problems into more narrowly defined, and thus more manageable, subproblems. Such architectures describe a set of components and their interactions, reflecting our current understanding of the problem space. In prior technical work, we have presented detailed reference architectures [16]. Here, we focus on highlighting the separation of concerns among three key components to help structure this brief overview.

First, an identification module that processes users' questions (Section 3.1.1). Second, a retrieval engine that accesses relevant documents (Section 3.1.2). Third, a discovery engine that orchestrates a strategy involving:

1. applying identification techniques to derive search criteria;
2. combining these criteria into a plan that leverages available indices; and
3. answering the question—often with user involvement.

We conclude the section with a set of examples that illustrate how these three components work in practice (Section 3.1.3).

3.1.1 Identification Interfaces

Data discovery solutions must offer a means for agents to express their information needs. For tabular data, identification interfaces include:

Keyword Search. Soon after keyword search became the standard interface for expressing information needs on the web, it was adapted to relational database management systems through early prototypes that had significant academic impact in the years that followed.

Query-by-Example (QBE). QBE interfaces were originally proposed as an alternative to SQL for querying relational databases [28]. The same idea—where agents provide examples of the desired output—has since been adapted for use in data discovery systems.

Task-Oriented. Instead of explicitly articulating the information need induced by a task, agents provide a direct specification of the task itself. When an evaluation function for the task is available, it can be supplied to the system, which then assumes responsibility for discovering data that satisfies the

[2]See, for example, [23] for surveys that summarize techniques used in discovery systems

task requirements. This approach was formalized for tabular data in Metam [13], with an earlier version for supervised machine learning tasks introduced in ARDA [4] and later applied in Kibana [20].

Personalization. When agent demand is high, systems can capture signals about what agents request and how satisfied they are with the results. These signals can then be used to build personalized models that better align with individual information needs. In web search, Google pioneered this approach, which has since become standard practice. In the context of tabular data discovery, certain data catalogs that capture *behavior*—the "B" in the ABCs of metadata [18]—lay the groundwork for more sophisticated personalization strategies.

3.1.2 Retrieval Strategies: Indices

It is useful to conceptualize the solution to the identification problem as producing a concrete search criterion. A retrieval strategy then takes this search criterion—such as a keyword, question, range, or spatial polygon—and uses an index to return matching documents. We define an index broadly as any data structure that facilitates the efficient location of documents that match a given search criterion.

Table Representation. Indices used in data discovery represent tables in ways that facilitate matching them against search criteria—and in doing so, they induce specific representations of the tables themselves. Several types of indices are commonly used. Full-text search indices represent each column of a table as a sparse vector within a vector space model. Vector indices use embedding models to encode entire tables or their components as dense vectors [1, 9, 24]. The design space for such vector representations is large and rapidly evolving, particularly with the emergence of retrieval-augmented generation (RAG) systems for tabular data, as we discuss later. Another approach is to represent data using graphs, which is common in knowledge bases that encode relationships between terms in the data.[3]

Data catalogs, or "databases of metadata," are widely used in industry. When discovery questions can be answered using metadata alone, these systems are effective in identifying relevant tables. Many data catalogs also incorporate behavioral signals derived from user interactions—corresponding to the "B" in the ABCs of metadata [18]—which further enhance their ability to support discovery.

The indices described above are compatible and often complementary. Many data discovery systems combine multiple indexing strategies to expand their ability to answer diverse queries. For example, modern data catalogs increasingly incorporate vector-based and graph-based techniques alongside traditional metadata indexing.

Building Blocks. When discovery questions require combining multiple documents, simple retrieval of individual tables is insufficient. To support more complex queries, a range of techniques—referred to here as building blocks—can be integrated with the indexing methods described above. Notable examples include indices that facilitate table joins, unions, and augmentations (conceptually, a combination of joining and unioning). Additionally, other building blocks enrich the metadata available to discovery systems by annotating columns and tables or by generating descriptive summaries.[4]

3.1.3 Examples

We present a representative, though not exhaustive, set of systems to illustrate the identification, retrieval, and discovery engine components discussed above.

Discover [19]. Discover accepts keywords as input—serving as its identification strategy—and returns a combination of tables from a given relational schema that contain those keywords (retrieval). The

[3]This is a vast design space, and a comprehensive treatment of knowledge bases for data discovery is beyond the scope of this paper.

[4]This task is referred in the literature as semantic annotation.

discovery engine constructs a compact answer by identifying a Steiner tree of join operations connecting the relevant tables.

Infogather [26]. Infogather takes an input table and returns an augmented version of it—adding rows or columns sourced from a collection of other tables. Its primary technical contribution is a retrieval index specifically designed to support such augmentation operations. The discovery engine implements algorithms to search this index and produce the final result.

Goods [17]. Goods takes keywords as input and returns relevant tables based on those keywords (identification). Its retrieval index encodes both the content of tables and rich metadata, including references from code (e.g., ETL and transformation pipelines) and user annotations. The discovery engine uses traditional information retrieval techniques to rank the results.

Aurum [7]. Aurum accepts programs written in a domain-specific language (DSL) that allows users to describe their information needs (identification). It uses a graph-based index to determine which subset of documents satisfies the DSL query. The discovery engine orchestrates the execution and combination of results as specified by the program.

LlamaIndex [22]. When equipped with the table reader plugin, LlamaIndex creates a vector representation for each row in the input tables (retrieval) and uses a vectorized form of a natural language question (identification) to find relevant tables. The discovery engine matches these vectors against the table representations and uses a large language model (LLM) to further refine the results.

3.2 My Group's Work

The diagram in Figure 2 presents our group's technical contributions to the area of data discovery and serves as a reference throughout this section to summarize key lessons learned. Before introducing the diagram, I offer a brief summary of the vision that has guided my work in this area from the outset.

Vision. There are countless compelling problems in data discovery. Working on any of them presents opportunities to contribute solutions that others can build upon, gradually advancing toward more complete systems. However, my philosophy has been that this piecemeal approach risks getting stuck in a kind of local optima—focusing on technically interesting subproblems that, while valuable, may not significantly advance the broader goal: helping people find the data they need.

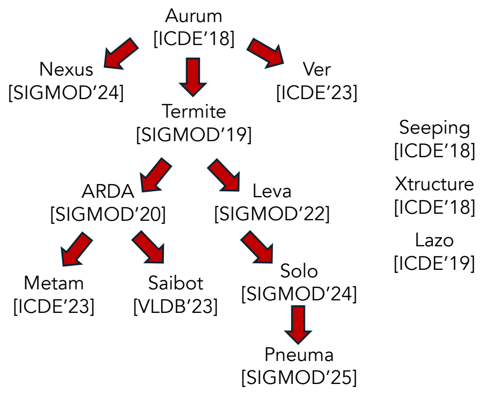

Figure 2: Portfolio of data discovery work

Over the past few years, I have prioritized an end-to-end approach to data discovery. This strategy comes with both drawbacks and advantages. One drawback is that building full prototypes takes considerable effort. Often, the resulting systems are not immediately deployable or impactful outside the lab—bridging that gap requires even more work. The longer development cycles also mean feedback loops can be slower.

The benefit, however, is substantial: pursuing end-to-end solutions reveals where the real bottlenecks lie. I use the term bottleneck informally to refer to two things i) technically interesting problems for which we lack good solutions, and ii) engineering challenges that differ in kind from those tackled by existing systems. Through this approach, I believe we have developed a strong sense of both.

3.2.1 Overview of the Group's Data Discovery Work

Our group's work is summarized in the diagram shown in Figure 2. Each node represents a technical paper, and edges indicate that one work builds upon another. While no summary can fully capture the richness and detail of these contributions, the diagram helps surface the main themes. We highlight those themes next, organized according to the reference architecture introduced earlier.

A Spectrum of Identification Techniques. When users have a clear idea of what they want, they can articulate their query using a domain-specific language (DSL) with systems like Aurum [7] or specify a utility function with Metam [13], the latter having introduced the concept of goal-driven data discovery. In situations where users cannot precisely define their information needs, alternative methods are available. For example, Aurum supports keyword search; Termite [9] offers vector search over tables and columns; and Ver [16] enables users to specify the desired table via a QBE-style interface. Finally, in systems such as Solo [25] and Pneuma [2], users express their information needs through natural language questions.

Table Representation. Supporting the identification strategies described above are a variety of table representation methods developed across our work. Seeping Semantics [10] was one of the first efforts to use distributed representations (embeddings) to capture the semantics of data in the context of data discovery. Termite extends this approach by embedding all values in a document, and Leva [27] advances the line further by exploring more effective techniques for computing embeddings at the table, column, row, and cell levels. Many of the insights from these efforts are incorporated into Solo and, more recently, Pneuma [2].

Complementing these vector-based approaches, systems such as Aurum [7], Nexus [14, 15], Xtructure [21], Metam [13], and Ver [5, 16] rely on sets of profiles to represent the underlying documents, offering a structured alternative to embedding-based representations.

Indices. Table representations are indexed using a variety of structures, depending on the system. Graph-based indices are used in Aurum, Ver, Metam, and ARDA [4]; vector databases are employed by Leva, Solo, and Pneuma; Lazo [11] uses hash-based indices; and Xtructure [21] relies on regular expression (regex) indices.

Broad Applications. Aurum, Seeping, and Ver support general-purpose discovery queries and offer the broadest interfaces among our systems, accommodating a wide range of information needs. In contrast, ARDA [4], Saibot [20], and Metam [13] focus on data augmentation. While ARDA and Saibot are tailored to supervised machine learning tasks, Metam generalizes this approach to any task for which a user can define a utility function.

Beginning with Leva, and continuing with Solo and Pneuma, we introduced RAG-like architectures that accept natural language questions and return relevant tables (i.e., table search). However, these systems do not support on-the-fly table assembly. Finally, Nexus targets hypothesis generation, where the hypothesis space is defined in terms of correlations among variables.

3.2.2 Tenets and Lessons Learned

Reflecting during and after the work described above has led to a number of insights that may be valuable to others working in this area.

Tenet 1: The North Star Is Satisfying Information Needs Directly. The overarching goal remains constant: help users find data that advances their work. This principle has guided many disciplines, though the techniques evolve. As systems improve, user expectations rise—pushing the target forward. This is a healthy dynamic, as long as we continue to learn how people want to access and use information, and we engineer systems that serve those real-world needs.

Tenet 2: Separate Identification from Retrieval. Clearly distinguishing between identification (understanding what data is needed) and retrieval (locating and delivering documents representing that data) is essential. This separation, rooted in software engineering principles like separation of concerns, helps both system design and research progress. Variants of this distinction have long existed in fields such as library science and information retrieval, albeit under different terminology. Adopting this lens helps bring conceptual clarity and makes it easier to incorporate insights from adjacent disciplines.

Tenet 3: Engineer for Evolving Signals. Data discovery does not have exact solutions; it resembles information retrieval more than deterministic SQL processing. As a result, systems must be engineered to flexibly incorporate new signals—reference datasets, human input, LLM prompts, knowledge graphs, and more. A guiding principle: avoid discarding documents early. Instead, cast a wide net, rank based on available signals, and update the ranking as new signals emerge. Always expose the basis for ranking to the user, and let them decide how to weigh or filter results.

Tenet 4: Favor End-to-End Solutions. As discussed earlier, pursuing end-to-end systems—rather than isolated point solutions—yields deeper insight into what actually limits progress. While point solutions are valuable (and abundant in the literature), end-to-end approaches reveal both technical and practical bottlenecks, and thus offer a clearer path toward usable systems. These efforts are more demanding, but ultimately more impactful.

Tenet 5: Be Cautious with Benchmarks. When benchmarks accurately reflect real-world problems, they are invaluable—they support reproducibility, enable progress tracking, and foster community alignment. But when a benchmark simplifies or misrepresents a problem, it can be misleading. In such cases, it encourages over-optimization on artificial targets, diverting attention from what actually matters in practice. Reproducibility is important, but not at the cost of chasing the wrong goals.

4 How do LLMs Change the Game?

Addressing data discovery requires solving both the identification and retrieval problems. Each involves a range of technical challenges—some of which remain especially difficult and unresolved.

- **Multiple Document Formats.** Understanding a table often requires access to contextual information—metadata that describes or qualifies the table's content—which is frequently not stored in tabular form. Much of this context resides in heterogeneous formats, including spreadsheets, PDFs, code, and other unstructured or semi-structured documents. If, as argued in Tenet 2 (Engineer for Evolving Signals), effective discovery depends on leveraging all available signals, then robust support for manipulating multiple document formats—or multimodal data—is essential.
- **Semantic Ambiguity.** Even with robust support for multiple document formats, resolving semantic ambiguity remains a core challenge. It requires determining which signals to prioritize in order to disambiguate meaning in specific contexts. Semantic ambiguity is pervasive in data discovery—from interpreting the unit of a value in a cell, to selecting an appropriate join key, to inferring a user's intent. These ambiguities often lack a single correct resolution, making adaptive, context-aware reasoning essential.
- **Scalability.** The value of data discovery increases as the volume of data grows—especially when manual exploration becomes impractical. At the same time, the imperative to incorporate new signals (as discussed in Tenet 2) continuously expands the set of documents under consideration. Processing large volumes of data—often requiring complex computations—poses significant scalability challenges, both in terms of system performance and engineering effort.

4.1 The Promise of LLMs

Large Language Models (LLMs) play a central role in many current research efforts. When conceptualized as vast repositories of documents equipped with natural language interfaces—and as systems capable of adapting their responses to specific information needs—they offer a compelling solution to the identification problem in data discovery. This makes LLMs a particularly interesting artifact to consider in the broader landscape of data discovery systems.

First, they readily provide approaches to tackle the first two problems above:

Querying Multimodal Data. Internally, LLMs represent all input, regardless of its original format, as vectors. These vector representations can be accessed and manipulated through natural language, effectively turning language into a Rosetta Stone-like interface for querying multimodal data. This makes LLMs a promising tool for addressing the challenge of working across heterogeneous document formats.

Addressing Semantic Ambiguity. LLMs encode knowledge from vast collections of documents and can be leveraged to resolve many forms of semantic ambiguity. While individual instances, such as interpreting a value's unit or understanding user intent, are often trivial for humans, the variety and unpredictability of such cases make it difficult to formalize a comprehensive set of rules. LLMs offer a compelling alternative: they can empirically address this long tail of ambiguities through contextual inference, without requiring explicit rule specification.

Second, LLMs can be leveraged both to help users articulate their information needs and to interpret those needs in ways that guide the system toward meaningful answers.

4.2 Challenges Ahead

Despite the promise, LLMs also introduce challenges: i) how to incorporate external data; ii) scalability.

Incorporating External Data. There are three broad approaches to incorporating external data, such as the document collections over which discovery queries are run, into LLM-based systems. First, external data can be incorporated during the training process, embedding the information directly into the model's parameters. Second, it can be provided as part of the input context at inference time. Third, external data can be accessed dynamically using a Retrieval-Augmented Generation (RAG) architecture, in which relevant documents are retrieved and passed to the model alongside the user's query.

Incorporating data during the training process, even through fine-tuning, is costly and raises concerns about data leakage, which may conflict with organizational priorities around privacy, compliance, and intellectual property. In contrast, in-context learning, RAG architectures, and self-play environments, where agents interact dynamically with data, offer more flexible and privacy-preserving alternatives. While these approaches have primarily been developed for unstructured data, our group and others are now extending them to tabular data with promising results, which we summarize next.

RAG and agentic architectures combine traditional systems for solving the retrieval problem with an LLM that receives both the question and partial retrieval results to generate an answer. This is where LLMs offer a distinct advantage: they help address the identification problem by interpreting the user's intent and connecting it to relevant data.

Despite promising early results, it remains too soon to determine the right RAG architecture for data discovery. However, we expect that the growing interest in this space will soon lead to practical and usable solutions—particularly if the scalability challenge, which we explain next, is addressed.

Scalability Challenge. While data discovery already faced scalability challenges before the advent of LLMs, those challenges are amplified by the use of LLMs and RAG architectures [25]. The primary source of this increased burden is the reliance on vector representations, which are both computationally expensive to generate—requiring resource-intensive model inference—and large in size, often spanning hundreds or thousands of dimensions. This makes it difficult not only to index data efficiently, but also to keep vector representations incrementally updated as the underlying data evolves. Despite these challenges, the substantial research activity and industry investment in this area suggest that practical solutions are likely to emerge.

Overall, LLMs represent an exciting and promising direction for data discovery. Given the substantial research and industry resources dedicated to overcoming their limitations, it is reasonable to expect significant progress in the coming years. However, the precise architectures and scalability mechanisms that will ultimately enable robust, real-world data discovery systems remain uncertain, highlighting the richness and complexity of the research landscape. Even once these technical challenges are addressed, the question of how fully they will resolve the broader data discovery problem remains open. In the next section, we offer some perspectives on what may still be missing.

5 Data Discovery is a Socio-Technical Problem

We believe that LLM-augmented data discovery architectures will, in the near future, meaningfully assist humans in identifying and retrieving documents that satisfy their information needs [8]. Yet, these systems will not fully resolve the data discovery problem. The stronger claim we make in this section is that no technical solution alone—regardless of how advanced—can fully solve data discovery.

The reason is subtle. For a technical solution to effectively help a user, two conditions must be met: the necessary information needs to be stored as documents, and the discovery tool they use must be able to locate those documents.

Up to now, data discovery research has mainly dealt with the second problem: finding and retrieving documents that satisfy a user's information need, often taking for granted that the necessary data is already in document format. We anticipate that near-future systems, like RAG and agentic systems, will begin to tackle this assumption. But the main takeaway from this section is that solving the first condition—getting the data represented as documents—is a tougher nut to crack without fully recognizing that data discovery is a socio-technical issue.

5.1 Data Discovery as a Socio-Technical Problem

The ultimate success of technical solutions hinges on the extent to which necessary data is captured in accessible documents. Often, much of the vital context needed to interpret, transform, and leverage these documents resides solely with data employees. Although these employees occasionally document datasets and processes, thereby generating valuable resources for technical systems, much of this knowledge remains undocumented. This is frequently due to the high cost of documentation, a lack of incentives (as it's often not considered part of their core responsibilities), or simply because the information is too recent to have been recorded.

To achieve data discovery in such scenarios, it's crucial to fully acknowledge the social component. This involves extracting the necessary data from the minds of data employees so that it can be converted into documents and then utilized by technical solutions. However, a number of reasons make this challenging:

- While we've characterized the identification process as something a single user undertakes, it's often the case that the "user" actually represents a team of individuals who must collectively decide what information is most important.
- Although human cognition is crucial to the identification process, our understanding of how these cognitive functions operate is currently insufficient to influence them beneficially.

Simply put, we need to make an effort to pinpoint what information users (or teams) actually need, and a separate effort to get the necessary documents from data employees who often store that data in their minds. Data discovery can't be solved without successfully connecting this supply of information with the demand. This realization was central to an article we wrote a few years back, where we introduced data markets as a solution for data discovery within organizations [12]. Since then, we've been building on these concepts through our research agenda, Data Ecology.

5.2 Data Ecology

Data ecology examines how documents are created and shared within data ecosystems, which are made up of different agents working to complete data-related tasks. This field proposes that by identifying all existing dataflows (the transfers of documents between agents), we can determine several things: which positive dataflows are already present and should be maintained; which beneficial dataflows are missing and should be encouraged; which negative dataflows exist and should be stopped; and which harmful dataflows don't yet exist and should be prevented from ever forming.

To achieve this control over dataflows, data ecology introduces tools for developing and evaluating *interventions*. These interventions are changes to the ecosystem designed to produce the desired dataflow effects. They can be technical (like the data escrow system we developed to manage data release and prevent leakage), but also economic (suchives as incentives), legal, and social (like establishing norms).

Understanding data discovery as a socio-technical issue and an organization as a data ecosystem to be analyzed via the dataflows that occur changes the game. A designer's objective then is to implement interventions that establish productive dataflows. This means encouraging data employees to document their *tribal knowledge* and helping data users articulate their information needs. Only when these pieces are in place can future data discovery systems effectively bridge the gap between data supply and demand.

The core idea is to foster an internal data ecosystem where data experts contribute their knowledge as documents to assist those who need to consume that data. In our previous paper, we explored mechanisms to encourage these dataflows within organizations, including gamification strategies like bonuses and monetization. We believe such internal data markets hold significant promise for addressing information needs at various levels, as previously illustrated in Figure 1. However, the design possibilities for these ecosystems are vast, and this remains an open problem. Our research efforts in recent years have primarily focused on developing the foundational concepts of Data Ecology.

6 Conclusions

Solving data discovery will significantly enhance data utility by lowering the costs associated with identifying and retrieving relevant documents, thereby accelerating progress in both science and industry. Historically, data discovery has been hampered by persistent challenges like semantic ambiguity and scalability, which are inherently difficult to overcome. However, with the rapid commoditization of large language models (LLMs) and our growing understanding of how to effectively integrate them with architectures like RAG, we now have access to a technology that could dramatically speed up our progress in data discovery. We are optimistic that the technical hurdles of data discovery will be largely resolved in the near future.

At the same time, we recognize that data discovery is fundamentally a socio-technical problem. Even if we had a perfect technical solution, we'd still face hurdles: helping users clearly define their information needs and convincing data providers to document the valuable knowledge they hold in their heads. To bridge this gap between data supply and demand, our Data Ecology research agenda has been exploring various data ecosystems, including internal data markets. While the technical challenges are still significant, we believe that highlighting the socio-technical dimension of this problem can greatly enhance the effectiveness of future solutions.

References

[1] Gilbert Badaro, Mohammed Saeed, and Paolo Papotti. 2023. Transformers for tabular data representation: A survey of models and applications. *Transactions of the Association for Computational Linguistics* 11 (2023), 227–249.

[2] Muhammad Imam Luthfi Balaka, David Alexander, Qiming Wang, Yue Gong, Adila Krisnadhi, and Raul Castro Fernandez. 2025. Pneuma: Leveraging LLMs for Tabular Data Representation and Retrieval in an End-to-End System. *Proceedings of the ACM on Managment of Data* (2025).

[3] Nicholas J Belkin. 1980. Anomalous states of knowledge as a basis for information retrieval. *Canadian journal of information science* 5, 1 (1980), 133–143.

[4] Nadiia Chepurko, Ryan Marcus, Emanuel Zgraggen, Raul Castro Fernandez, Tim Kraska, and David Karger. 2020. ARDA: automatic relational data augmentation for machine learning. *arXiv preprint arXiv:2003.09758* (2020).

[5] Kevin Dharmawan, Chirag A Kawediya, Yue Gong, Zaki Indra Yudhistira, Zhiru Zhu, Sainyam Galhotra, Adila Alfa Krisnadhi, and Raul Castro Fernandez. 2024. Demonstration of Ver: View Discovery in the Wild. In *Companion of the 2024 International Conference on Management of Data*. 428–431.

[6] Raul Castro Fernandez. 2025. What is the Value of Data?: A Theory and Systematization. *ACM/IMS Journal of Data Science* (2025).

[7] Raul Castro Fernandez, Ziawasch Abedjan, Famien Koko, Gina Yuan, Samuel Madden, and Michael Stonebraker. 2018. Aurum: A data discovery system. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. IEEE, 1001–1012.

[8] Raul Castro Fernandez, Aaron J Elmore, Michael J Franklin, Sanjay Krishnan, and Chenhao Tan. 2023. How large language models will disrupt data management. *Proceedings of the VLDB Endowment* 16, 11 (2023), 3302–3309.

[9] Raul Castro Fernandez and Samuel Madden. 2019. Termite: a system for tunneling through heterogeneous data. In *Proceedings of the Second International Workshop on Exploiting Artificial Intelligence Techniques for Data Management*. 1–8.

[10] Raul Castro Fernandez, Essam Mansour, Abdulhakim A Qahtan, Ahmed Elmagarmid, Ihab Ilyas, Samuel Madden, Mourad Ouzzani, Michael Stonebraker, and Nan Tang. 2018. Seeping semantics: Linking datasets using word embeddings for data discovery. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. IEEE, 989–1000.

[11] Raul Castro Fernandez, Jisoo Min, Demitri Nava, and Samuel Madden. 2019. Lazo: A cardinality-based method for coupled estimation of jaccard similarity and containment. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. IEEE, 1190–1201.

[12] Raul Castro Fernandez, Pranav Subramaniam, and Michael J Franklin. 2020. Data market platforms: trading data assets to solve data problems. *Proceedings of the VLDB Endowment* 13, 12 (2020), 1933–1947.

[13] Sainyam Galhotra, Yue Gong, and Raul Castro Fernandez. 2023. Metam: Goal-oriented data discovery. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*. IEEE, 2780–2793.

[14] Yue Gong and Raul Castro Fernandez. 2024. Demonstrating Nexus for Correlation Discovery over Collections of Spatio-Temporal Tabular Data. In *Companion of the 2024 International Conference on Management of Data*. 524–527.

[15] Yue Gong, Sainyam Galhotra, and Raul Castro Fernandez. 2024. Nexus: Correlation Discovery over Collections of Spatio-Temporal Tabular Data. *Proceedings of the ACM on Management of Data* 2, 3 (2024), 1–28.

[16] Yue Gong, Zhiru Zhu, Sainyam Galhotra, and Raul Castro Fernandez. 2023. Ver: View discovery in the wild. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*. IEEE, 503–516.

[17] Alon Halevy, Flip Korn, Natalya F Noy, Christopher Olston, Neoklis Polyzotis, Sudip Roy, and Steven Euijong Whang. 2016. Goods: Organizing google's datasets. In *Proceedings of the 2016 International Conference on Management of Data*. 795–806.

[18] Joseph M Hellerstein, Vikram Sreekanti, Joseph E Gonzalez, James Dalton, Akon Dey, Sreyashi Nag, Krishna Ramachandran, Sudhanshu Arora, Arka Bhattacharyya, Shirshanka Das, et al. 2017. Ground: A Data Context Service.. In *CIDR*. Citeseer.

[19] Vagelis Hristidis and Yannis Papakonstantinou. 2002. Discover: Keyword search in relational databases. In *VLDB'02: Proceedings of the 28th International Conference on Very Large Databases*. Elsevier, 670–681.

[20] Zezhou Huang, Jiaxiang Liu, Daniel Gbenga Alabi, Raul Castro Fernandez, and Eugene Wu. 2023. Saibot: A Differentially Private Data Search Platform. *Proceedings of the VLDB Endowment* 16, 11 (2023), 3057–3070.

[21] Andrew Ilyas, Joana MF da Trindade, Raul Castro Fernandez, and Samuel Madden. 2018. Extracting syntactical patterns from databases. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. IEEE, 41–52.

[22] llamaindex. [n. d.]. Llamaindex. `https://www.llamaindex.ai`

[23] Norman W Paton, Jiaoyan Chen, and Zhenyu Wu. 2023. Dataset discovery and exploration: A survey. *Comput. Surveys* 56, 4 (2023), 1–37.

[24] Liane Vogel, Benjamin Hilprecht, and Carsten Binnig. 2023. Towards foundation models for relational databases [vision paper]. *arXiv preprint arXiv:2305.15321* (2023).

[25] Qiming Wang and Raul Castro Fernandez. 2023. Solo: Data Discovery Using Natural Language Questions Via A Self-Supervised Approach. *Proceedings of the ACM on Management of Data* 1, 4 (2023), 1–27.

[26] Mohamed Yakout, Kris Ganjam, Kaushik Chakrabarti, and Surajit Chaudhuri. 2012. Infogather: entity augmentation and attribute discovery by holistic matching with web tables. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*. 97–108.

[27] Zixuan Zhao and Raul Castro Fernandez. 2022. Leva: Boosting machine learning performance with relational embedding data augmentation. In *Proceedings of the 2022 International Conference on Management of Data*. 1504–1517.

[28] Moshe M. Zloof. 1977. Query-by-example: A data base language. *IBM systems Journal* 16, 4 (1977), 324–343.