Towards Semantic Exploration of Tables in Scientific Documents

Varish Mulwad [1], Vijay S. Kumar [2], Jenny Weisenberg Williams [2], Tim Finin [3], Sharad Dixit [2] and Anupam Joshi [3]

[1] *GE Research, John F. Welch Technology Center, Whitefield, Bengaluru, India*
[2] *GE Research, 1 Research Circle, Niskayuna, NY, USA*
[3] *University of Maryland Baltimore County, 1000 Hilltop Circle, Baltimore, MD, USA*

Abstract

Structured data artifacts such as tables are widely used in scientific literature to organize and concisely communicate important statistical information. Discovering relevant information in these tables remains a significant challenge owing to their structural heterogeneity, dense and often implicit semantics, and diffuse context. This paper describes how we leverage semantic technologies to enable technical experts to search and explore tabular data embedded within scientific documents. We present a system for the on-demand construction of knowledge graphs representing scientific tables (drawn from online scholarly articles hosted by PubMed Central), and for synthesizing tabular responses to semantic search requests against such graphs. We discuss key differentiators in our overall approach, including a two-stage semantic table interpretation that relies on an extensive structural and syntactic characterization of scientific tables, and a prototype knowledge discovery engine that uses automatically-inferred semantics of scientific tables to serve search requests by potentially fusing information from multiple tables on the fly. We evaluate our system on a real-world dataset of approximately 120,000 tables extracted from over 62,000 COVID-19-related scientific articles.

Keywords

Scientific Tables, Table Characterization, Semantic Table Interpretation, Knowledge Graphs, Semantic Search

1. Introduction

The discovery of relevant information from scientific documents has traditionally been critical to the advancement of our understanding of real-world phenomena. The pace of research in dynamic, fast-evolving scenarios, as exemplified by COVID-19, can lead to unprecedented rates in publication of scholarly literature on a given subject [1]. Furthermore, recent advances in generative AI and foundation models have made it radically simpler and more economical for the scientific community to produce and publish new technical content [2]. These emerging trends have necessitated more machine-driven, human-interpretable approaches to scientific knowledge discovery. Open datasets like CORD-19 [3] have motivated novel techniques to enable scientists to comprehend existing literature, including tools for keyword/semantic search and Q&A, recommendation, and summarization of scientific documents. Newer models [4] and tools like ChatPDF [5] and Elicit [6] foster dialog-driven information discovery from documents within various domains. However, as with web documents, the exploration of scientific literature is predominantly associated with searching over its unstructured textual content.

Besides text, alternative modalities such as tables and charts play a significant role in how the scientific community conveys descriptive information in the literature. Our experience with assembling a corpus of over 62,000 open-access coronavirus-related articles from PubMed Central [7] yielded over 120,000 tables underlining a wealth of latent knowledge embedded within such structured data artifacts. Extraction and retrieval of relevant information from these scientific tables is becoming increasingly

Proceedings of the ESWC 2023 Workshops and Tutorials; 1st International Workshop on SemTech4STLD, May 28, 2023, Hersonissos, Greece
EMAIL: varish.mulwad@ge.com (V. Mulwad)—*Corresponding author*; v.kumar1@ge.com (V. S. Kumar); weisenje@ge.com (J. W. Williams); finin@umbc.edu (T. Finin); sharad.dixit@ge.com (S. Dixit); joshi@umbc.edu (A. Joshi)
ORCID: 0000-0001-9113-5952 (Mulwad); 0000-0003-2234-1546 (Kumar); 0000-0002-6593-1792 (Finin); 0000-0002-8641-3193 (Joshi)

critical to emerging knowledge-driven applications. For example, consider a genomic surveillance scenario seeking information on *treatment efficacies against the top prevalent COVID-19 variants in each US state*. Better responses to such queries may entail going beyond text and searching relevant portions of or entire scientific tables for vital knowledge nuggets, possibly fusing information from multiple source tables on the fly.

Viewing scientific documents as a collection of content-rich tables and descriptive context, this paper addresses the need for technical experts to search and systematically explore scientific tables in their quest to discover new knowledge. In comparison to traditional web tables and open relational data, tables found in scientific documents are generally (i) smaller in size, (ii) consist of cells with more idiomatic strings or numerical data than text and (iii) contain specialized textual entities specific to a scientific domain (see Figure 1). Moreover, unlike their relational/web table counterparts, the nature of scientific tables reflects the more general circumstances of their creation: When researchers prepare scientific manuscripts, their contents are typically subject to constraints in terms of "publication real estate" (e.g., maximum page count for a paper). Consequently, especially with any tables present in these articles, researchers resort to numerous information compaction practices to optimize for human consumption and visual perception of information in the tables while ensuring that there is no ambiguity, information overload or other loss in informational value associated with these tables.

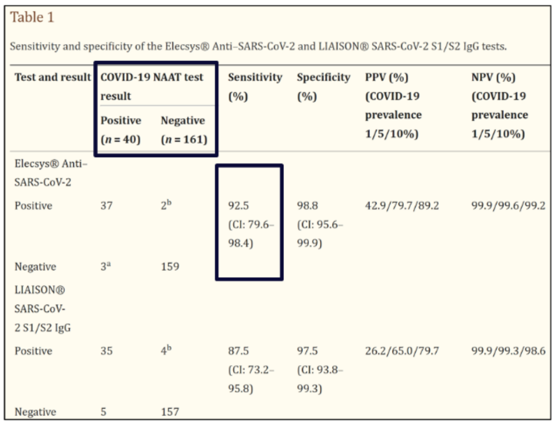

Table 1

Sensitivity and specificity of the Elecsys® Anti–SARS-CoV-2 and LIAISON® SARS-CoV-2 S1/S2 IgG tests.

Test and result	COVID-19 NAAT test result		Sensitivity (%)	Specificity (%)	PPV (%) (COVID-19 prevalence 1/5/10%)	NPV (%) (COVID-19 prevalence 1/5/10%)
	Positive (*n* = 40)	Negative (*n* = 161)				
Elecsys® Anti–SARS-CoV-2						
Positive	37	2[b]	92.5 (CI: 79.6–98.4)	98.8 (CI: 95.6–99.9)	42.9/79.7/89.2	99.9/99.6/99.2
Negative	3[a]	159				
LIAISON® SARS-CoV-2 S1/S2 IgG						
Positive	35	4[b]	87.5 (CI: 73.2–95.8)	97.5 (CI: 93.8–99.3)	26.2/65.0/79.7	99.9/99.3/98.6
Negative	5	157				

Less text, more numbers … sub-columns … merged cells

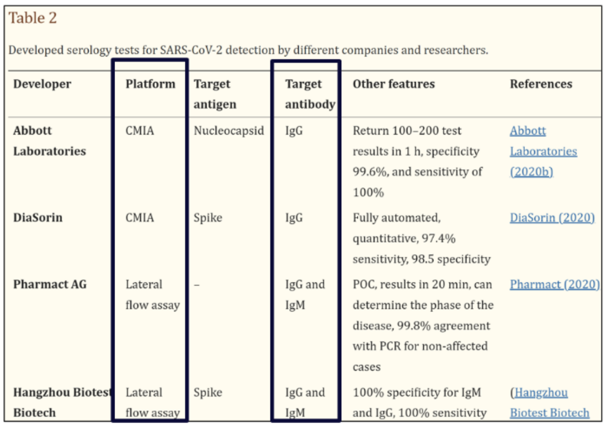

Table 2

Developed serology tests for SARS-CoV-2 detection by different companies and researchers.

Developer	Platform	Target antigen	Target antibody	Other features	References
Abbott Laboratories	CMIA	Nucleocapsid	IgG	Return 100–200 test results in 1 h, specificity 99.6%, and sensitivity of 100%	Abbott Laboratories (2020b)
DiaSorin	CMIA	Spike	IgG	Fully automated, quantitative, 97.4% sensitivity, 98.5 specificity	DiaSorin (2020)
Pharmact AG	Lateral flow assay	–	IgG and IgM	POC, results in 20 min, can determine the phase of the disease, 99.8% agreement with PCR for non-affected cases	Pharmact (2020)
Hangzhou Biotest Biotech	Lateral flow assay	Spike	IgG and IgM	100% specificity for IgM and IgG, 100% sensitivity	(Hangzhou Biotest Biotech

Similar to web tables … with domain-specific entities

Figure 1: Example snippets from real tables appearing in PubMed Central articles (*PMC7456296*, *PMC7694563*) that exhibit some of the typical characteristics of tables in scientific documents.

Our work is motivated by the resulting distinctive characteristics of scientific tables listed below:

- **Structural heterogeneity**: In the absence of universal guidelines, tables in scientific documents often come in a variety of structures and dimensions. To better fit tabular data within the available space, researchers may (i) transform a simple table with only column headers into a more complex one with row headers or both sets of headers, (ii) merge neighboring cells with repeated values into one, or (iii) group logically related neighboring cells and extract any common information into higher-level parent cells—potentially resulting in a hierarchy of sub-rows and/or sub-columns.
- **Dense and often implicit semantics:** Due to space constraints, researchers may (i) use abridged table header names whose contextual description may explicitly capture dense semantics—e.g., a header cell *mort. rt.* to signify mortality rate could either represent deaths per 1000 of infected population or deaths per million of the total population, (ii) coalesce numerical quantities and their associated units within the same table cells, or (iii) use headers where context is implicit, i.e., the meaning of a header cell (e.g., '*location*'), and hence of an associated column or row, can be ascertained only by inspecting the content of data cells in that column or row.
- **Diffuse Context:** Tables in scientific papers typically represent a summary view or a meta-analysis over a more comprehensive set of information. (i) Contextual description of cell content may be present in table captions or other referring text outside of the tables, (ii) Cells may include pointers to other related tables instead of actual tabular data themselves. From a discovery viewpoint, it is likely that information needed to put together a comprehensive response to a search request is scattered across multiple tables and sources, potentially drawn from different technical documents.

The very practices seeking to ease human understanding of published scientific tables, thus, also introduce challenges that make it harder for machine-driven table understanding and discovery. As part of our approach to address challenges posed by structural heterogeneity and domain-specific entities (e.g., gene sequences, any measurements/units that hold a special meaning within a scientific domain), we model scientific tables as linked data objects and construct knowledge graphs (KG) using information extracted from these tables and metadata from their respective containing articles. In this paper, we present a KG-creation system that employs a two-stage table interpretation methodology. For each table, we first perform both an extensive characterization of the table structure as well as a syntactic understanding of its individual cell contents. Next, semantic table interpretation, focusing on column type annotation (CTA) and cell entity annotation (CEA), leverages the results of the first stage to map table cells to nodes in an existing reference KG (e.g., Wikidata [8]). Semantic interpretation of web tables is widely understood – While we build on our practical solution adapted to scientific tables in the biomedical domain [9], it is not our key contribution. Our overall approach for scientific tables can leverage existing alternative web table interpretation systems with appropriate adaptations. For exploring our scientific tables-centric-KG, we additionally present a hybrid table-based semantic search paradigm to support table-matching based on inferred semantics under various contextual constraints.

Our specific contributions described in this paper include: **(1)** a dataset collection system to harvest tables and provenance metadata from online open-access PubMed Central articles, **(2)** a preliminary prototype system to generate a KG of scientific tables that includes ***a)*** annotation of the collected tables along multiple dimensions using automated rule-based structural characterization and syntactic parsing techniques, and ***b)*** knowledge graph embeddings-based approach to semantic table interpretation, **(3)** a promising preliminary evaluation over a subset of annotated data and a thorough characterization of over 120,000 scientific tables that demonstrate high structural heterogeneity, **(4)** a pipeline to populate RDF triples representing semantic annotations of table cells and table provenance in a KG, and **(5)** a prototype discovery engine that leverages the inferred table semantics captured in the KG to support discovery of tabular data by potentially merging semantically compatible tables on the fly. This paper presents our first steps towards populating and exploring a KG of scientific tables. We are in the process of open sourcing our current technology for the community to reproduce and build on our work[2].

2. Related Work

Web Tables and Dataset Search: Given their ubiquity in communicating information, an extensive body of prior research by the information retrieval and database communities covers the processing and understanding of tables, particularly around inferring the intended meaning and semantics of both web tables and Open Data for dataset search and discovery purposes [10][11][12][13]. Some approaches allow table-based search and data enrichment where queries take the form of an input table—schema matching is then performed to discover ‘unionable’ or ‘joinable’ tables that are ranked based on some similarity criteria [12] to generate on-the-fly result tables [14]. However, existing approaches do not specifically address the idiosyncrasies of tables found in scientific and technical documents.

Table Representation Learning: Recently, the use of large-scale, pretrained table representation learning (TRL) models [15][16] for understanding and exploring tabular information has been gaining traction across communities, with a few efforts even specifically addressing understanding of tables found in scientific documents. Challenge datasets like PubTables-1M [17], ChemTables [18], ArxivPapers [19], and SciGen [20]—all assembled from different forms of scientific/technical literature—are fostering innovation in table detection, classification, extraction, context-aware table-to-text generation, and question-answering respectively. Complementary to these, and from a discovery perspective, our work targets on-the-fly synthesis of tabular knowledge from existing scientific tables.

Semantic Table Interpretation: The SemTab challenge [21] annually benchmarks semantic table interpretation systems with a focus on relational web tables (scientific tables from biomedical papers were included in recent editions). SemTab systems aim to map table data to nodes in a KG such as Wikidata or DBpedia. Specifically, they map a column header string to a KG class (Column Type

[2] As of submission, our code to assemble scientific tables from PMC articles on-demand is available under: https://github.com/orgs/ge-knowledge-discovery/repositories. Code for the remaining systems will be made available here pending code-release approvals.

Annotation (CTA)), data cell string to a KG entity (Cell Entity Annotation (CEA)), and relationship between pair of columns to a KG property (Column Property Annotation (CPA)). Existing systems [22][23][24] could potentially be adapted for CTA and CEA in scientific tables to replace our current approach. Beyond SemTab, semantic interpretation of scientific tables has seen limited research—[25] leverages the MetaMap tool to annotate cells in PMC article tables with UMLS concepts. QTLTableMiner++ [26] annotates tables in Europe PMC articles with concepts from Phenotype, Plant, and other quantitative trait locus-specific ontologies. Our work goes further combining structural and syntactical characterizations with a table's semantic interpretation (via CTA and CEA) to generate a more comprehensive semantic representation to support table search including on-the-fly data fusion.

Scientific Table Characterization: Existing techniques to formally characterize scientific literature [27] cover many aspects of scholarly knowledge but are limited in their representation of tables found in these documents. [28] presents a taxonomy for web tables, categorizing them into "relational knowledge" tables (those from which knowledge can be extracted) and "layout" tables (no knowledge to extract). Another taxonomy [29], henceforth referred to as the Lautert taxonomy, classifies web tables primarily into five mutually exclusive categories (horizontal, vertical, matrix, formatting, navigational), mirroring [28], but omits subcategories that are unlikely to be encountered in scientific documents (e.g., form, calendar), and uses more common names (e.g., "horizontal table" instead of "vertical listing table"). The secondary classification includes five non-mutually exclusive categories (concise, nested, multivalued simple, multivalued complex, splitted), each indicating a different methodology to structurally format and present tabular content. Fang et al. [30] introduce additional labels (e.g., multi-level header, multi-dimensional table) that are applicable to scientific tables but are not adequately covered in Lautert's scheme. To our knowledge, only Milosevic et al. [31][32] have developed a system for structural characterization of tables in PMC articles. Their rule-based system detects and distinguishes between header and data cells, detects basic data types for cells (e.g., strings, numbers), and classifies a table into one of three types. Our proposed approach, in comparison, can generate characterization at different levels (cell, row, column, table), generating values for 18 different labels.

3. Dataset Collection

We developed an extensible, reusable data collection pipeline to download, parse, and extract a dataset of tables and associated metadata from COVID-19-related PMC manuscripts. Inspired by the CORD-19 dataset [3], we instantiate our pipeline with the following query search-string: ("COVID" OR "COVID-19" OR "Coronavirus" OR "Corona virus" OR "2019-nCoV"). As of 06/26/2021, this query resulted in over 200,000 matching PMC articles, of which 62,777 articles were present within the PMC open-access, commercial-use collection subset. Further, of these, **62,746 articles** contained corresponding full-text encodings in XML format. Upon parsing these XML artifacts, our pipeline was able to extract a total of **120,417 HTML-formatted tables** from across the article corpus.

4. A Knowledge Graph of Scientific Tables

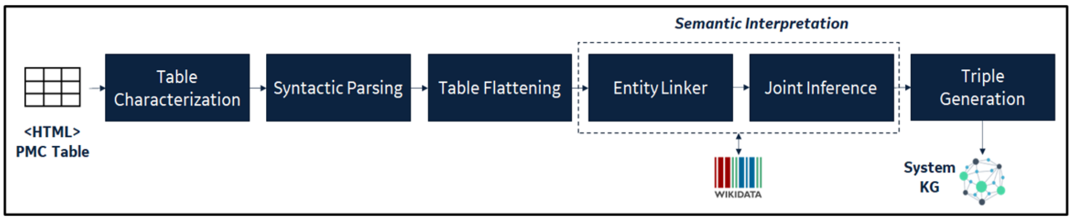

Figure 2: Our KG generation system infers structural and syntactic characteristics along with semantic interpretation of tables in scientific documents

Figure 2 shows an overview of our system to semantically interpret tables discovered in scientific literature. The HTML tables extracted by the dataset collection pipeline are first processed by a **Table Characterization** module that infers the different structural and syntactic characteristics at the cell, row, column, and table level, including the basic datatypes (e.g., numbers, strings) of table cells. The table is further processed by the **Syntactic Parsing** module which infers additional higher-level syntactic and semantic types (e.g., Quantity, DNA/RNA sequence) and separates the core (query) string from its contextual metadata (e.g., units). Together, these two modules play a key role in the process of semantic interpretation of tables in scientific literature. The information inferred by these modules helps – i) untangle the structural heterogeneity and diffuse context thereby enabling the generation of accurate RDF triples, and ii) extract only those core cells/rows/columns of the table that can be effectively mapped to KG nodes and separate them from those ones that should not be (e.g., literals). As we show in our evaluations, unlike web tables, a significant fraction of tables in scientific documents are composed of primarily literals as opposed to linkable concepts & entities. A subset of tables is further processed by the **Table Flattening** module which converts complex structures into simpler relational-like tables. While optional, its benefits are twofold: i) easier triple generation and ii) the ability to leverage joint inference approaches to collectively map headers and cells to KG nodes.

The **Semantic Interpretation** module follows, what is now a standard approach for inferring table semantics, by mapping header cells to concepts and data cells to entities in an existing KG (in our case, Wikidata [8]). It additionally comprises a joint inference component, which, whenever possible, collectively assigns entities and concepts to a header and cells in a column. The inferred structural, syntactic, and semantic knowledge along with the document metadata is represented in the form of RDF triples and populated into a KG via a **Triple Generation** module.

4.1. Table Characterization

The table characterization module leverages several rules to identify different characteristics at the cell, header, row, and table level. We first describe the taxonomy used to identify these characteristics followed by our implementation details. To categorize tables, we start with the Lautert taxonomy [29] and supplement it with additional classifications from [30]. Our characterization also includes additional labels and metrics (e.g., number of rows, number of columns, basic data types for table cell values, semantic types for certain set of commonly encoded data), which are generally useful and/or needed to derive the Lautert [29] and Fang [30] classifications.

Table 1
Labels to characterize scientific table structure

Label	Level	Type
MAIN_CLASSIFICATION	table	Enum
NUM_HEADER_ROWS	table	Number
NUM_BODY_ROWS	table	Number
NUM_COLS	table	Number
HEADER_ROWS_MULTILEVEL	table	Boolean
HEADER_ROWS_CONCISE	table	Boolean
BODY_CONCISE	table	Boolean
HAS_HEADER_ROW	table	Boolean
HAS_HEADER_COLUMN	table	Boolean
HAS_MULTIVALUED_CELLS	table	Boolean
CELLS_DATATYPE	table, row, col	Enum
NUM_CELLS	row, col	Number
HAS_EMPTY	row	Boolean
HAS_ROWSPAN	row	Boolean
HAS_COLSPAN	row	Boolean
BOLD	col	Boolean

We identified and selected several labels that can be applied at the cell, row, column, header, and table level. At the table level, a main classification label can be one of **Horizontal**, **Vertical**, **Matrix**, or **Multi-dimensional**. The header level characterization labels include **Header Row**, **Header Column**, **Header Row Concise**, **Header Row Multilevel**, and **Headers Splitted**. The table body (minus header cells) is characterized to be concise or not (**Body Concise**). Cell level labels include checks for whether a cell is multi-valued (further propagated up to the table level) and also several basic data types like string, number, number with range, etc.

Implementation: We implement a rule–based table characterization module which assigns 18 labels applied at the row, column, or table label, as seen in Table 1. Internally, the algorithm may apply some of these labels at the cell level as well. Labels may have one of three types of values: (1) number, e.g., number of columns in a table (2) Boolean, e.g., whether a table has multilevel header rows (3) enumeration, e.g., a given row's data type. Some labels indicate that a table (or row/column) contains a feature (e.g., HAS_COLSPAN=True), whereas some labels indicate that the table (or row/column) is comprised entirely of the given feature (e.g., CELLS_DATATYPE="number"). The implementation generates labels using an incremental approach largely leveraging the HTML markup of the tables. In general, rows and columns are characterized based upon inspection of their component cells. Likewise, tables are characterized based upon inspection of their component rows and columns. The characterization module follows a strict order driven by logical dependencies—it cannot determine if a table is horizontal before characterizing the datatype of its columns; it will not characterize columns before determining that a table is non-concise. Such dependencies drive how labels are calculated.

An illustrative example follows, showing a subset of the module's logic leading up to classifying Table 1 from article PMC5923605 (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5923605) as a horizontal table (meaning that each row contains information for an entity). From the table's HTML markup, the module determines that it contains two body rows, and that for each row, no CELL_DATATYPE can be assigned since it contains a mix of strings, numbers, and numbers with tolerances. Further, the module determines that no body rows contain an HTML element (rowspan/colspan) causing a cell to exceed the height or width of a single row or column, and therefore classifies the table as BODY_CONCISE=false (does not contain merged cells). This classification indicates that the table is a regular grid, and therefore the module can characterize its columns. It determines that there are five columns, and that each has a consistent datatype (two columns are CELL_DATATYPE="string", two columns are CELL_DATATYPE="number", and one column is CELL_DATATYPE="number with tolerance"). Note that if the column was mostly a certain datatype but contained cells indicating missing values (whitespace, dash, or "NA"), then it will still correctly assign the datatype. Finally, the module determines that since the table contains at least two columns with distinct and differing datatypes (omitting the first column, which is often a header), it assigns MAIN_CLASSIFICATION="horizontal". In a horizontal table, each row represents an entity, and each column represents a characteristic of that entity. In this example table, the entities are the samples, and the columns contain phases, particle sizes, densities, etc. of the sample.

4.2. Syntactic Parsing & Table Flattening

Once a table's structural and syntactic characteristics have been identified, we further process it to simplify the semantic interpretation step. This includes identifying commonly encoded data in tables and separating the core (query/entity) string from its contextual metadata. The former assists in identifying a subset of the table (cells/rows/columns) that should not be semantically annotated while the latter produces string content per cell for direct use by the semantic interpretation module.

Detecting & Labelling Commonly Encoded Data: Accurately interpreting table semantics requires not only linking strings to appropriate concepts/entities from a reference KG, but also not linking strings/numbers that represent literals. While literals found in scientific tables can be classified into a few basic types (e.g., strings, numbers, Boolean values, dates), it is useful to classify them into more specific types. For tables in the biomedical domain, examples of such specific types include DNA/RNA sequences, references, and clinical trial IDs. Similar to [33], we develop a specialist-based framework for detecting commonly encoded data in tables. A "specialist"—either a regular expression/pattern-based, dictionary-based, or machine learning approach, independently assesses cells in a column/row for the likelihood that the column belongs to a specific type that it is designed for. For example, there could be independent specialists to detect zip codes, SSNs, and sequences. We implement specialists to detect two basic types and three semantic types (Table 2). These were identified by inspecting common entries in the tables in our dataset. Cells/rows/columns identified as one of these types do not require any additional semantic interpretation. The syntactic parsing module uses each specialist to determine the basic and/or semantic type for each cell. Further, it propagates this value to the column/row (header) if all cells in a column/row have the same type (ignoring cells with empty entries, "NA", or dashes).

Table Flattening: A final step before semantic interpretation involves normalizing or flattening a subset of tables with "complex" structures. We leverage the inferred table characterizations to identify tables with a single simple header row and concise body containing only row spans. We flatten such tables using specialized data libraries (e.g., pandas) and re-export table HTML filling missing entries.

Table 2
List of "specialists" and how we implemented them in our system

Basic Type: **Date**	Pattern-based approach to recognize dates
Basic Type: **Quantity**	Use the Python package Pint (https://pint.readthedocs.io/en/stable) to recognize strings representing magnitude and units (e.g., 20 ft.)
Semantic Type: **DNA_RNA_SEQUENCE**	Pattern-based approach to detect cells comprised of strings matching 3 or more of {G, A, T, C, U}
Semantic Type: **CLINICAL_TRIAL_ID**	Pattern-based approach to detect strings of the format NCT* (e.g., NCT12345678)
Semantic Type: **REFERENCE**	Cells referencing works cited in an article; detected by the presence of an HTML <`xref`> tag containing attribute `ref-type`=*"bibr"*

4.3. Semantic Interpretation

We implement our table interpretation module for scientific literature with a focus on CTA and CEA [21], mapping header and data cell strings to Wikidata [8] items (i.e., concepts and entities). At the core of our module is a scalable entity linker designed and optimized for the biomedical domain [9]. Additionally, our module includes a joint inference component, which, wherever applicable, collectively assigns entities to header and all data cells in a column.

Core Entity Linker: We developed a practical, scalable entity linker that takes a mention string (from a table cell) and retrieves ranked lists of top k Wikidata items that it deems to be best matches to that mention string. In [9], we describe how we customized this linker for the biomedical domain and optimized the linking process for hundreds of thousands of table cells by setting up a localized instance of the Wikidata knowledge graph—specifically, indexing 95.8M items and the type hierarchy comprising 2.6M types. We use the 'sitelinks' count for each item as a proxy for its popularity.

Joint Inference: The core entity linker uses limited context while generating and ranking candidate entities for table cell strings. In a well-formed (simple/flattened) table, there are several relations/interactions between its constituent header and data cells, which can be leveraged to disambiguate entity assignments for table cells. These include (1) semantic type of headers influencing the entities assigned to data cells in a column and vice-versa; (2) semantic type of a header influencing and being influenced by the semantic types of adjoining headers; and (3) entity assignment of a data cells influencing and being influenced by the assignments of adjoining data cells in a row [24]. We implement a joint inference component specifically focusing on collectively assigning entities to cells in a table column. At the core of this component is a KG embeddings-based "agreement function" to compute compatibility between the entities assigned to data cells in a column. KG embeddings [23] are vector representations of nodes in a knowledge graph. We leverage existing pre-trained embeddings for Wikidata provided via Wembedder [34]. It provides both pre-trained models as well as REST APIs for accessing embeddings for Wikidata items and properties.

The algorithm starts with the ranked list of candidates retrieved by the core linker for each data cell in a column. For each candidate, we use the Wembedder API to retrieve its corresponding embedding vector (if available). Once the vectors for all the candidates for all cells are retrieved, we perform K-means clustering over the collection of vectors, producing a cluster assignment prediction for each candidate vector. At this point, each cell is associated with an ordered list of candidate entities with corresponding vectors and cluster IDs. The algorithm identifies the cluster ID ("mode cluster ID") most frequently associated with the top-ranked vectors across the cells. For each cell, the algorithm selects the highest-ranked candidate entity that is assigned to the mode cluster ID. If no candidate entities are assigned to the mode cluster ID, then the algorithm picks the first candidate entity for which a vector was available. If none of the candidate entities have associated vectors, then it selects the first-ranked candidate. The K-means clustering algorithm requires a key input indicating the target number of clusters. The joint inference algorithm cannot be applied to all tables - it works best when our system is able to identify a column in a flattened table. In cases where joint inference is not applicable, the module chooses the top-ranked candidate returned by the core linker.

4.4. Evaluations

Table Characteristics Evaluations: We select 8 table-level labels likely to be most useful for the downstream table flattening and semantic interpretation modules. We selected PMC [7] articles at random from our dataset and manually annotated their tables with values for these 8 labels. For each label, we calculate precision (of the labels assigned by the system, what percentage are correct) and recall (of the labels expected to be assigned by the system, what percentage are assigned). Results are shown in Table 3. The "# Sys." column indicates the number of tables characterized by the module with each of the labels across the entire set of over 120,000 tables (e.g., Tables with Concise Headers =36,182 indicates the table characterization module inferred that 36,182 tables have a concise header).

Table 3
Table Characterization is evaluated across 8 labels

Tables with …	# Sys.	# Anno.	Pr.	Re.
Header rows	113,582	110	1.00	0.94
Header columns	48,733	103	1.00	0.55
Concise header rows	36,182	34	0.84	0.94
Multi-level header rows	32,169	33	1.00	0.97
Only numeric data cells	12,969	29	1.00	0.83
Concise body	40,158	39	0.97	0.67
Horizontal orientation	21,863	38	0.95	0.50
Vertical orientation	7,205	16	0.91	0.62

The column "# Anno." indicates the number of tables annotated for use in evaluating the characterization accuracy for each label. Precision ("Pr.") for all labels is relatively high, indicating that if the system generates a label, it is likely to be correct. We observed lower recall ("Re.") for labels related to concise body, header columns and main classification (related to table orientation), indicating areas where the system fails to detect an expected label. It is often challenging to detect a concise table body because a table's appearance often does not match its underlying formatting (specifically, a cell that appears to be merged is often formatted as a set of cells, some of which are empty). The system often misses identifying horizontal and vertical tables if it has no way to differentiate column or row data types (e.g., if every column is a number), or if it cannot characterize columns due to the table being concise. It is challenging to identify header columns if the algorithm cannot detect anything different about the header column (e.g., all bold, or the only string column in the table).

Syntactic Parsing and Semantic Interpretation Evaluations: We manually annotated 47 tables drawn at random from 45 PMC articles. Each table cell was mapped to one of: Wikidata item/property (henceforth referred to as Wikidata), Literal, Reference, Clinical Trial IDs, Index (header cells representing index columns in a table), or NA (indicating that it should be mapped to a concept/entity, but none exists in the reference KG). NA was also used in a small fraction of cases where the table cell composed of multiple entities (e.g., "*Maine, Ohio, California*" as single cell value). A total of 3600 table cells were annotated. Unlike web tables where most cells represent entities, in our dataset, a majority of the table cells (~**70%**) are literals, followed by Wikidata annotations (~**25%**) as the second-most frequent type. We use this annotated data to evaluate the syntactic parsing and semantic interpretation modules' capabilities. Joint inference was evaluated using Wembedder's pre-trained model [34] over a June 2017 snapshot of Wikidata, NUM_CLUSTERS parameter set to 4, and # of initial candidates retrieved by the core entity linker set to 25.

Table 4 shows the breakdown for precision (Pr.), recall (Re.), and F-score for the different types predicted by the system. We skip reporting on NA and index as they are not supported by our system yet. Rule-based methods to detect basic types, and specialist-identified semantic types achieve high precision for detecting Literals, Clinical Trial IDs, and References. Though, as is typical with rule-based methods, recall gets impacted if patterns do not have sufficient coverage (as evidenced in the case of Trial ID). The numbers for the semantic interpretation module to produce accurate Wikidata links appear lower on the surface.

Table 4
Distribution of manual annotation and system performance across different types

Anno. Type	# of cells	Pr.	Re.	F-score
Literal	2,548	0.98	0.81	0.89
Wikidata	910	0.33	0.50	0.40
NA	118	-	-	-
Reference	14	0.91	0.71	0.80
Index	6	-	-	-
Clinical Trial ID	4	1.00	0.50	0.67

We first analyze the candidate set of Wikidata items generated by the core entity linker. Of the 910 table cells manually annotated with Wikidata items, the linker is able to retrieve the correct expected annotation in the top 25 candidates for 554 cells (~**60%**), while it is unable to for the remaining 356 cells (~**40%**). If we consider the subset of 554 cells for which the expected annotation was retrieved in the candidate set, the Wikidata **recall increases** to 458/554 (**0.82**). The lower precision for Wikidata can be analyzed by looking into the 1373 table cells for which the system produced Wikidata links. The expected annotation for 488 of those cells is either literals, reference, index, NA, or clinical trial IDs which the linker mischaracterizes as Wikidata items. Out of the remaining 885 cells with expected Wikidata annotations, the system accurately links 458 cells. If we consider this subset, **precision increases** to 458/885 (**0.51**). Using an older version of Wikidata embeddings also arguably has an impact on the overall accuracy.

4.5. Triple Generation and Knowledge Graph Construction

To model tables in scientific documents and their annotations, we developed an ontology to represent metadata from PMC documents along with the data extracted from their constituent tables and their inferred semantic meaning. This ontology builds upon a model [35] which, in turn, extends a subset of the standard W3C PROV ontology. We subclass the core `ENTITY` class from PROV and extend it to represent PMC documents and tables. The Triple Generation module takes the output from the Table Characterization and Semantic Interpretation modules and represents the data as RDF triples grounded in our ontology. Specifically, it includes a pipeline that auto-generates triples as follows: First, it creates triples at the document level (e.g., title) followed by creation and linking of the document to instances representing its authors, publishers, and sponsors. Next, it creates triples for tables in the document, starting at the table level (e.g., number of rows) down to the cell level (e.g., parsing minimum and maximum numeric values from a cell containing a range). It then adds triples for the Wikidata assignment for cells as returned by the Semantic Interpretation module. The number of triples generated per document varies widely according to its contents but is on the order of 1000 per PMC article. These triples are persisted in a knowledge graph for table discovery purposes.

5. Search and Discovery of Scientific Tables

We developed a framework for enhanced search and discovery of relevant information across scientific tables captured in a knowledge graph, and for organizing this information in the form of tabular responses. This section presents a preliminary evaluation of a prototype discovery engine that uses the inferred semantics of tables to serve table-based semantic search requests.

Hybrid Table-based Semantic Search: When searching over tables in scientific documents, a meaningful response is often predicated on greater query expressivity and semantic resolution of user requests than can be provided by a simple keyword search. To this end, we provide a hybrid search experience that supplements keyword search with the ability for users to figuratively 'sketch' their desired (relational) result table—including specifying semantic markups of its header/data cells, along with constraints around its contextual metadata. This ensuing table-based semantic search request is systematically translated into queries to retrieve appropriate data from the knowledge graph.

We first parse any table-based discovery request into an intermediate query plan where both tables and operations on tables are first-class citizens. Inspired by relational database systems, our query plans are composed of primitive operators (`SELECT`, `FILTER`, `RANK`, and `FUSE`) to process scientific tables expressed as linked data objects in a knowledge graph. The SELECT operator simply returns a list of identifiers for all tables in our graph that match with the discovery request. The FILTER operator prunes this list by applying one or more contextual constraints on the tables. RANK orthogonally reorders the list of tables based on some ranking criteria.

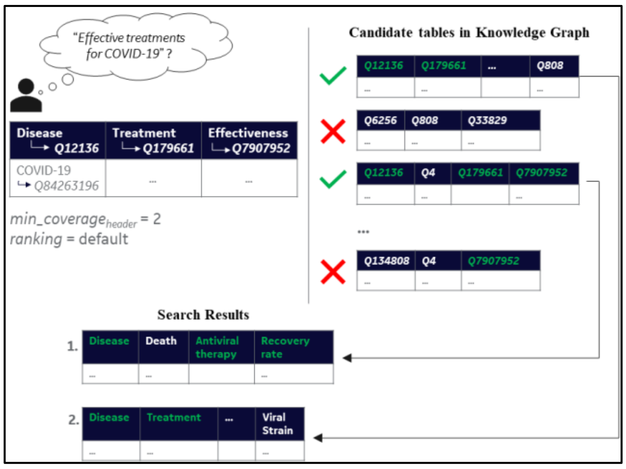

Figure 3. Using Wikidata QIDs of header cells to guide discovery engine operations

Discovery Engine and Semantics-guided Table Discovery: We built a prototype discovery engine that takes as input an intermediate query plan and incrementally constructs and executes appropriate SPARQL queries against a knowledge graph of scientific tables. This is accomplished by adding or modifying ad hoc query clauses corresponding to each operator instance in a query plan. This engine is also responsible for taking the results of SPARQL query execution and packaging them in the form of relational result tables for easy consumption.

Discovery of relevant tabular information involves matching the semantics of a list of search terms from a user request against the appropriate inferred semantics of tables and filtering the matches to include only those tables that meet some similarity criteria. Consider a simplified search request around latest findings on effective treatments for COVID-19 (see Figure 3). The matching algorithm underlying the SELECT operator takes the semantics (in this example, Wikidata identifiers or QIDs) associated with header cells in the table-based search request and compares them against those associated with header cells for each table in the graph. If the two sets of QIDs overlap completely, or if the set overlap exceeds a threshold as established by some query constraint (e.g., min_coverage=2), then the candidate table is included in the search results. By default, RANK sorts the list of candidates based on this header-cell coverage metric (i.e., tables with maximum set overlap are ranked higher).

Evaluation: We evaluated the ability of our discovery engine to return a ranked list of relevant tables corresponding to a search request – here, a request comprises a desired list of entities that a result table header is expected to contain, and relevancy is based on a semantic match between this list and the header cells of tables. Specifically, we carried out a preliminary evaluation against a knowledge graph comprising a subset of 118 tables drawn at random from our full set of tables, and manually examined for relevance to a query set. Our query set includes 5 queries motivated by realistic scenarios, where each query is a list of QIDs corresponding to three desired entities. We compute Average Precision (**AP**): the mean of precision scores at each point where a relevant result is returned, and Recall: the ratio of relevant results to the expected number of relevant results based on a manual search. We also compute Mean Average Precision (**MAP**) scores across all queries.

Table 5
Performance of semantics-guided table discovery on ranked retrieval tasks

Metric	Result size (# tables)	Query 1: Q12136 Q134808 Q18420531	Query 2: Q12136 Q134808 Q186408	Query 3: Q33829 Q65096341 Q2221906	Query 4: Q6256 Q92307203 Q55215251	Query 5: Q6256 Q92307203 Q41792217	MAP
AP	5	0.5	0.33	0.83	1	0.7	0.67
	10	0.5	0.33	0.83	0.67	0.61	0.59
	25	0.5	0.33	0.83	0.5	0.55	0.54
Recall	5	1	1	0.67	0.33	0.25	
	10	1	1	0.67	0.67	0.625	
	25	1	1	0.67	1	1	

As seen in Table 5, overall precision scores are relatively low across all queries, whereas all queries demonstrate high recall. We are exploring hybrid approaches to better match against header cells, including going beyond strict QID matches to also compare against a hierarchy of an entities' super-types, and semantic similarity based on pre-trained Wikidata embeddings from Wembedder [34]. With scientific tables, the ideal response to a search request is one that merges relevant information from tables potentially drawn from multiple documents. To this end, our engine also supports a `FUSE` operator that can union the rows of two or more semantically-compatible candidate tables on the fly and generate a fused table in response to the request (details beyond the scope of this paper).

6. Conclusions

The discovery of relevant information from scientific documents, till date, has predominantly focused on unstructured text, ignoring crucial information encoded in structured artifacts such as tables. In this paper, we demonstrated preliminary prototype systems for populating and searching over a knowledge graph of scientific tables, enabling semantics-guided discovery from these structured artifacts. Our two-stage table interpretation methodology highlighted the vast structural and syntactical heterogeneity prevalent in scientific tables by characterizing over 120,000 tables from PMC articles. Preliminary evaluations of the characterization and semantic interpretation modules on a subset of these tables show promising results. While the characterization is a high precision module, it's recall could be improved for certain labels by including additional rules or exploring the development of supervised algorithms. Semantic interpretation and disambiguation performance is good when our system is able to discover the appropriate entity in its ranked candidate result set but fails otherwise. Candidate search needs to be further adapted to the domain strings' idiomatic nature. Entity linking systems have traditionally focused on linking strings to nodes in a knowledge graph, but in scientific tables—where a majority of the cells are literals—they also need to detect and learn to not link literals. Distinguishing between a literal and an entity string will be a key driver for this. Our preliminary evaluation of table discovery, based solely on strict header-based semantic matching, showed low average precision but high recall. Enhanced semantic matching of tabular content and the system's ability to fuse knowledge from multiple tables on the fly will drive a successful discovery experience over scientific documents in the future. In this paper, we believe we have taken initial steps towards developing a scalable and practical scientific table-centric knowledge graph population and search system.

7. Acknowledgements

This research is based on work supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via [2021-21022600004]. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, or the U.S. Government.

8. References

[1] Holly Else, How a Torrent of COVID Science changed Research Publishing—in seven charts, Nature, Vol. 588 (553), 2020. doi: 10.1038/d41586-020-03564-y.
[2] Mark Carrigan, Generative AI and the Unceasing Acceleration of Academic Writing, 2023. URL: https://blogs.lse.ac.uk/impactofsocialsciences/2023/03/14/generative-ai-and-the-unceasing-acceleration-of-academic-writing/ (last accessed: April 2023).
[3] L. L. Wang. K. Lo, Y. Chandrasekhar, R. Reas, et al., CORD-19: The COVID-19 Open Research Dataset, in: Proceedings of Workshop on NLP for COVID-19 at ACL Conference, 2020.
[4] R. Luo, et al., BioGPT: Generative Pre-trained Transformer for Biomedical Text Generation and Mining, Briefings in Bioinformatics, 23(6), 2022. doi: 10.1093/bib/bbac409.
[5] ChatPDF—Chat with any PDF!, 2023, URL: https://www.chatpdf.com/ (last accessed: April 2023)
[6] Ought, Elicit: The AI Research Assistant, 2022. URL: https://elicit.org (last accessed: April 2023)

[7] PMC Open Access Subset, Bethesda, MD: National Library of Medicine, 2003. https://www.ncbi.nlm.nih.gov/pmc/tools/openftlist/ (last accessed: April 2023)
[8] Denny Vrandecic and Markus Krotzsch, Wikidata: A Free Collaborative Knowledgebase, Communications of the ACM, Vol. 57(10), pp. 78-85, 2014. doi: 10.1145/2629489.
[9] V. Mulwad, et al., A Practical Entity Linking System for Tables in Scientific Literature, in: Proceedings of the 3rd AAAI Workshop on Scientific Document Understanding, 2023. To appear.
[10] M. Cafarella, A. Halevy, H. Lee, J. Madhavan, C. Yu, D. Wang, E. Wu, Ten Years of WebTables, in: Proceedings of VLDB Endowment, 11(12): 2140-2149, 2018. doi: 10.14778/3229863.3240492
[11] Shuo Zhang and Krisztian Balog, Web Table Extraction, Retrieval and Augmentation: A Survey, ACM Trans. on Intelligent Systems and Technology, 11(2): 1-35, 2020. doi: 10.1145/3372117.
[12] A. Chapman, E. Simperl, L. Koesten, G. Konstantinidis, L. Ibanez, E. Kacprzak, P. Groth, Dataset Search: A Survey, The VLDB Journal, 29, pp. 251-272, 2020. doi: 10.1007/s00778-019-00564-x
[13] N. Noy, et al., Google Dataset Search: Building a Search Engine for Datasets in an Open Web Ecosystem, in: Proceedings of the 28th World Wide Web Conference, pp: 1365-75, 2019.
[14] S. Zhang and K. Balog, On-the-fly Table Generation, in: Proceedings of SIGIR, pp: 595-604, 2018.
[15] G. Badaro, M. Saeed and P. Papotti, Transformers for Tabular Data Representation: A Survey of Models and Applications, Eurecom Technical Report 6721, 2021.
[16] A Comprehensive Paper List of Reasoning over Tables, 2022, URL: https://github.com/yilunzhao/Awsome-Table-Reasoning (last accessed: April 2023)
[17] B. Smock, R. Pesala, R. Abraham, PubTables-1M: Towards Comprehensive Table Extraction from Unstructured Documents, in: Proceedings of CVPR, 2022. doi: 10.1109/CVPR52688.2022.00459.
[18] Z. Zhai, et. al., ChemTables: A Dataset for Semantic Classification on Tables in Chemical Patents, Journal of Cheminformatics, 13(97), 2021. doi: 10.1186/s13321-021-00568-2.
[19] M. Kardas, et al., AxCell: Automatic Extraction of Results from Machine Learning Papers, in: Proceedings of EMNLP, pp. 8580-8594, 2020. doi: 10.18653/v1/2020.emnlp-main.692.
[20] N. S. Moosavi, et al., SciGen: A Dataset for Reasoning-Aware Text Generation from Scientific Tables, in Proceedings of 35th Conference on NeurIPS, Track on Datasets and Benchmarks, 2021.
[21] N. Abdelmageed, et al., Results of SemTab 2022, in: Vol. 3320 of CEUR Workshop Proceedings —Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab), 2022.
[22] J. Liu, et al., From tabular data to knowledge graphs: A survey of semantic table interpretation tasks and methods, Journal of Web Semantics, Vol. 76, 2023. doi: 10.1016/j.websem.2022.100761
[23] J. Liu, et al., Radar Station: Using KG Embeddings for Semantic Table Interpretation and Entity Disambiguation, in Proceedings of ISWC, pp. 498-515, 2022. doi:10.1007/978-3-031-19433-7_29
[24] V. Mulwad, T. Finin, A. Joshi, Semantic message passing for generating linked data from tables, in: Proceedings of ISWC 2013, LNCS vol. 8218, doi: 10.1007/978-3-642-41335-3_23.
[25] N. Milosevic, et al.: A framework for information extraction from tables in biomedical literature. Intl. Journal on Document Analysis and Recognition (IJDAR) 22(1), 55–78, 2019.
[26] G. Singh, et al.: Qtltableminer++: semantic mining of qtl tables in scientific articles, BMC Bioinformatics, 19(183), 1–11, 2018. doi: 10.1186/s12859-018-2165-7.
[27] O. Corcho, A review of ontologies for describing scholarly and scientific documents, 2014.
[28] E. Crestan, P. Pantel, Web-scale table census and classification, in Proceedings of 4th ACM Intl. Conf. on WSDM. pp. 545–554, 2011. doi: 10.1145/1935826.1935904.
[29] L. R. Lautert, M. M. Scheidt, C. F. Dorneles, Web Table Taxonomy and Formalization, ACM SIGMOD Record 42(3), 28–33, 2013. doi: 10.1145/2536669.2536674.
[30] J. Fang, et al., Table header detection and classification, in: Proceedings of 26th AAAI Conference on Artificial Intelligence, 26(1), pp. 599-605, 2012. doi: 10.1609/aaai.v26i1.8206.
[31] N. Milosevic, et al., Disentangling the Structure of Tables in Scientific Literature, in: Proceedings of NLDB, LNCS vol. 9612, pp. 162-174, 2016. doi: 10.1007/978-3-319-41754-7_14.
[32] N. Milosevic, et al., Extracting patient data from tables in clinical literature—case study on extraction of bmi, weight and number of patients, in: HEALTHINF. pp. 223–228, 2016.
[33] P. Nimbalkar, et al., Semantic interpretation of structured log files, in: Proceedings of 17th Intl. Conf. on Information Reuse and Integration (IRI). pp. 549–555, 2016. doi: 10.1109/IRI.2016.81.
[34] F. A. Nielsen, Wembedder: Wikidata Entity Embedding Web Service, Preprint, arXiv:1710.04099.
[35] Provenance Modeling in RACK, 2021. URL: https://github.com/ge-high-assurance/RACK/wiki/RACK-Data-Model#provenance-modeling-in-rack (last accessed: 04/23)