Data Driven Design as an Ecologically Valid Challenge Task for Document Understanding

Sireesh Gururaja[1] Junwon Seo[2] Hung-Yi Lin[2] Jeremiah Milbauer[1]
Anthony Rollett[2] Emma Strubell[1,2]
[1]Language Technologies Institute, School of Computer Science
[2]Department of Materials Science and Engineering
Carnegie Mellon University
sgururaj@cs.cmu.edu

Abstract

In this work we present a benchmark dataset for zero-shot multimodal document understanding to support the data-driven design (DDD) of materials. The benchmark repurposes manually-verified, machine extracted textual and tabular data from two previously published DDD papers. The proposed dataset requires information extraction across rich document layouts, resolution of in-paper symbolic references, and numerical reasoning for normalization of quantitative data, in both multimodal and text-only settings. We argue that data-driven design presents a promising task — data-rich, useful, and challenging — against which to benchmark next-generation document understanding systems.

1 Introduction

Data-driven design (DDD), a process by which materials scientists use information extracted from the literature to inform future experiments, has emerged in the past decade as an important method by which to accelerate the discovery of materials (Olivetti et al., 2020). As NLP methods have evolved, so too has their application to data-driven design problems, from pipeline-based approaches using multiple purpose-trained models and relying heavily on rules-based, handwritten heuristics (Kim et al., 2017; Court and Cole, 2018; Jensen et al., 2019, *inter alia*) to end-to-end approaches involving fine-tuning large language models (LLMs) to act as information extractors and assistants (Zheng et al., 2023), or generate structured output describing properties directly (Dagdelen et al., 2024).

However, even current, LLM-based data-driven design work relies on laboriously collected annotated data. The method proposed in Dagdelen et al. (2024), for instance, suggests annotating “ 100–500 text passages” in order to fine-tune an LLM to produce structured data. This type of data can be difficult to produce: it often requires domain expertise to collect, verify, and postprocess into a format that is appropriate for training such models. This problem is exacerbated when considering that data-driven design efforts often seek to extract information into specific, non-overlapping schemas, limiting the possibility of data sharing or transfer learning between separate DDD efforts.

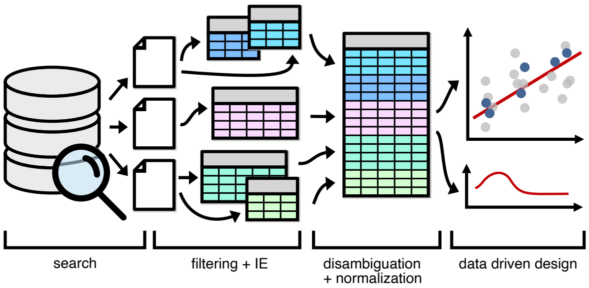

Figure 1: The process of data-driven design. Our benchmark focuses on the middle two phases: extraction/filtering and disambiguation/normalization

Given, however, the rapid development of models that can process scientific documents, both in text-only and multimodal formats, we view the possibility of data-driven design projects that require little to no annotated data as both highly desirable and feasible in the near future. The extraction challenges created by typical data-driven design projects also remain at the frontier of the capabilities of even newer models: processing visually-rich documents with information in text, tables, and figures; disambiguating extracted information to a standardized schema; and performing consistent numerical reasoning to normalize scales and units so that extracted information is comparable across papers (Miret and Krishnan, 2024).

In this paper, we propose a dataset to demonstrate DDD’s suitability as a challenge task and benchmark for next-generation document understanding models, focused on replicating a subset of the data derived from two prior DDD works: Jensen et al. (2019), which focuses on zeolite synthesis, and Pfeiffer et al. (2022), on aluminum alloys. Both of these datasets focuses on a different materials system, and the relevant information from each paper and the schema into which they are extracted are also different.

We intend for this benchmark to reflect a realistic subset of the data-driven design process, which often relies on downloading papers from publishers, which are typically provided in XML/HTML or PDF format. We therefore present two settings for this benchmark: a multimodal setting, which presents as input a full paper PDF (or a series of page images, for models that do not accept PDFs directly), and a text-only setting, in which input is XML/HTML. In both cases, the expected output is the disambiguated, normalized information from the corrected versions of the original dataset. We propose zero-shot baselines in both settings, and find that while modern systems perform strongly on common formats of tables, their ability to extract and integrate information varies widely between different sources of information and different table layouts.

Because we cannot republish the content of papers used in this benchmark, we release a script to reconstruct the dataset from the metadata provided in each source paper alongside evaluation code for the benchmark in our repository[1]

2 Data Driven Design and Task Scoping

In keeping with the literature, we conceptualize of DDD as a task separated into four phases, which we visualize in figure 1, and discuss below:

Search/retrieval In this phase, researchers typically collect a large number of papers using high-recall, low-precision methods like keyword matching. Papers are typically downloaded in a number of different formats, including scraped HTML and XML and PDFs from publisher APIs, then converted to text.

Information extraction and filtering In this phase, researchers will attempt to extract information corresponding to the schema of interest from the retrieved papers. Notably, in this phase, not all extracted information is relevant, necessitating a filtering process. The specific methods by which this phase is carried out have varied over time. Olivetti et al. (2020) describe an pipelined approach common at the time; end-to-end approaches have since become more popular.

Disambiguation and normalization In this phase, researchers attempt to make information extracted from retrieved papers comparable. This can be seen as a two-step process: disambiguating extracted information into the intended schema, and normalizing numerical values to be comparable, in both scale and units.

Visualization and Modeling The goal of data-driven design projects is typically not just the extraction and disambiguation of information, but using it to visualize existing literature, in order to plan future experiments, or to serve as a preliminary screen for promising new candidate materials by predicting properties of interest, such as in Zhang et al. (2024).

We argue that a useful evaluation for document understanding systems is to focus on the second and third phases and their associated tasks, namely information extraction, filtering, disambiguation and normalization. With this scope, we aim to present a system with the content of a paper and the desired schema, and have it output normalized information from the paper in that schema. Systems that perform well at this evaluation would be immensely useful to materials scientists: given a collected set of potentially useful papers and a desired schema, the system could automate the construction of a dataset that allows the modeling and prediction of potentially valuable new materials systems.

2.1 Task Settings

One of our primary focuses in designing this benchmark is *ecological validity*: which would imply that models that are successful at this benchmark would be able to be applied to real-world DDD tasks. For our benchmark to accurately reflect DDD as it is currently carried out, the models that we evaluate must be able to operate effectively on all of the formats in which publishers make their documents available. In our case, we collect PDF and XML/HTML documents from publishers through their APIs, according to our institutions' licensing agreements. We present two settings: a PDF setting and a text-only setting. In the PDF setting, the model receives as input the full PDF of the document, or alternatively a series of PNG images of each page if it does not process PDF files directly.

[1] https://anonymous.4open.science/r/ddd-benchmark-C3B8/README.md

In the text-only setting, the content of the XML document is provided. In neither case do we apply additional preprocessing to the documents being processed. We note that because most publishers do not provide data in both formats, these two settings are not directly comparable; to compensate, we highlight results from the set of papers that we do have available in multiple formats alongside the PDF-only and XML-only results. We envision the setup of this task to be that a model receives a set of input document, and a target schema into which to extract and normalize information, and produces a table in that schema as a result. Preprocessing, prompting, and in-context learning and training a model before the task are all considered parts of acceptable implementations for this task, but fine-tuning per-dataset would not be acceptable.

3 Dataset Construction

We begin the construction of our dataset from two prior data-driven studies: Jensen et al. (2019) and Pfeiffer et al. (2022). In this section, we first describe the process of collecting the paper content from which both studies extracted data, and then the further annotation and collection processes for both datasets. We finally detail the challenges that this dataset presents to models models in order to be successful at this task.

3.1 Data Collection and Distribution

To collect data across both modalities for this dataset, we focused on three publishers with relatively permissive text and data mining (TDM) licenses, for which our institution had an agreement in place. After resolving publisher metadata for each dataset with the CrossRef API,[2] we used publisher APIs from Elsevier and Wiley to download full-text XML and PDF files, respectively, and used the Springer integration with CrossRef to get PDF and XML/HTML if available. We focused on the use of TDM APIs such that this dataset can be replicated at any institution with comparable licensing, because we cannot redistribute the papers directly. Further, framing the task as applying models directly to the outputs of TDM APIs allows models successful at this task to be drop-in augmentations for new or existing DDD projects. For institutions that have less permissive licensing, we tailor our evaluation script to evaluate and compare candidate models with our baselines on subsets of the papers used for the baseline evaluations. See the discussion in 3.4.1. A limitation of this dataset is the scale of the data that is available to us because of the intersection of using manually checked and corrected data and complying with licensing terms. We argue, however, that because of the many-dimensional nature of the information being extracted, that this dataset is still useful as a benchmark.

3.1.1 Zeolites (Derived from Jensen et al. (2019))

The original zeolite dataset [3] consists of synthesis parameters and derived products of zeolites, a class of materials with many commercial applications. This paper extracts 1,638 rows of manually verified data from 116 individual papers, looking at content in tables, text, and supplementary information using a combination of learned and rule-based extraction. Zeolite synthesis typically involves creating a gel from several components: the elements that form the crystal, such as silicon and germanium, additional reaction components, such as water, and an organic molecule that directs the crystal formation. This dataset contains 12 columns of these ingredients, as well as several more that represent further normalization of their contents, or the results of corroborating simulations.

For our dataset, we focus on columns that are directly extracted from the papers themselves, removing derived columns. Because publishers often do not provide a way to programmatically access supplementary information, we additionally scope down our dataset to information derived from tables and text in the main paper. With our licensing constraints, we have a total of 55 papers, four of which are unavailable through Wiley's API. This results in 51 papers, 22 in PDF format, 39 in XML, and 10 in both formats. Our final dataset contains 414 rows of data, with a total of 4950/4968 non-null values. We provide an example table from this dataset in 3, along with a worked example of a subset of the extracted data in Appendix D.

3.1.2 Aluminum Alloys (Derived from Pfeiffer et al. (2022))

Pfeiffer et al. (2022) compiled a dataset of 1278 entries on mechanical properties of aluminum alloys extracted from tables. Because of the limitations of automated extraction tools, this was presented

[2] https://crossref.org

[3] Available at: https://github.com/olivettigroup/table_extractor/blob/master/zeolite_data/ge_synthesis_data.csv

as a separate dataset from the compositions that had those mechanical properties. While this data was validated against documented handbook values for common alloys, only outliers were manually checked. Pfeiffer et al. (2022) additionally note that many of the high strength outliers were due to the aluminum alloy being subject to severe plastic deformation (SPD) processing and were flagged but not removed.

In order to create the benchmark dataset all of the original entries were manually inspected by multiple researchers. There were additional entries that were related to SPD processes that were not flagged in the original dataset. Further there were several other categories of issues with extracted data which stray from the purpose of assessing properties of (industrially relevant/viable) aluminum alloys: many entries were mechanical properties of welds (particularly friction stir welds) or aluminum-based composites. The entries that were properties related to SPD, welds, or composites were removed from the dataset. Entries that were references to other works (e.g. referencing a handbook or earlier literature as benchmarks) were also removed.

Once the desired subset of entries from the original mechanical properties dataset was identified, additional information was added to each entry regarding the composition and the processing information, thus unifying the two separate datasets presented in Pfeiffer et al. (2022). Room temperature tempering steps, e,g, natural aging, are described at 25°C. All temperatures are in Celsius, all times are in hours. Compositions are in weight percent. This results in 8 columns related to composition and properties, and 28 columns that are weight fractions of individual elements. With our licensing restrictions, we were able to obtain 152 total papers, with 22 in PDF format, 151 in XML format, and 21 in both. This resulted in 330 rows, and 3806/12210 non-null values.

3.2 Diagnostic Datasets

In order to provide more detail on where models succeed and fail, we additionally annotate subsets of each dataset with location and layout information that indicates where information in each dataset, at a column granularity is found, and where it is expressed. For each datapoint, we annotate if the data was found in the text of the paper, or in several configurations of table. This comprises eight categories in three buckets: (1) Data from the tables (entire columns for that data, information in headers, or information in particular cells under hierarchical indices); (2) Data from text, even if linked to a table (generally text on the page, but also footnotes and table captions); or (3) not present in the paper.

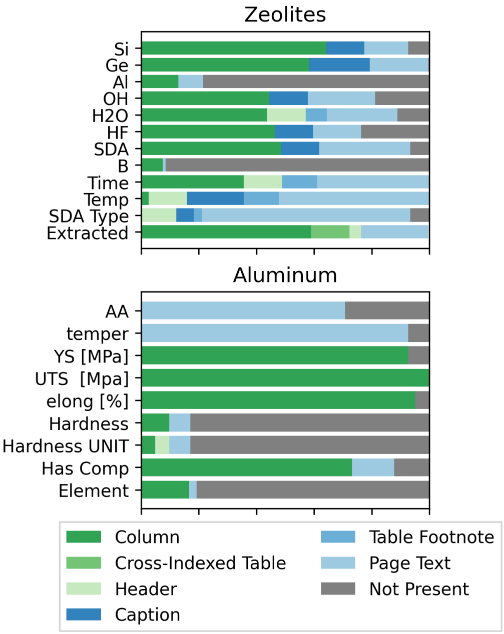

Figure 2: Distribution of data locations in the dataset per column type in each dataset. Green bars indicate information found within tables, blue indicates related text, and gray indicates absent information. Note that the aluminum dataset has much more homogenous sources of information.

We annotate data from 28 papers found in the zeolite dataset, and 40 papers from the aluminum dataset. We present a visualization of how data are distributed into these buckets in Figure 2.

3.3 Task Features

This task presents a number of interesting challenges to information extraction methods. In this section, we discuss these features, using an example of table parsing taken from Lorgouilloux et al. (2009, Figure 3).

Identifying Relevant Information. This task presents an example of long-context information extraction, where models are expected to extract information into a provided schema given the context of an entire scientific paper (which is often longer than the longest context window available), in which relevant text and tables have to be identified before being tagged.

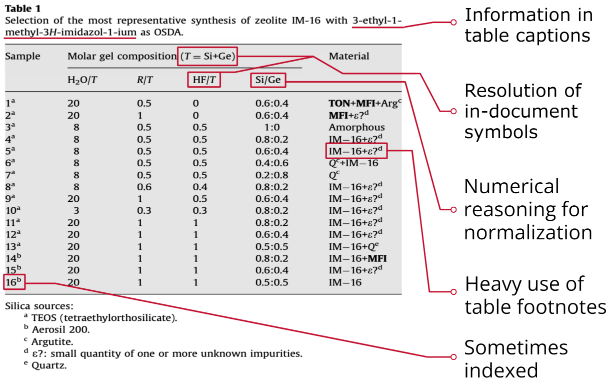

Table 1
Selection of the most representative synthesis of zeolite IM-16 with 3-ethyl-1-methyl-3*H*-imidazol-1-ium as OSDA.

Sample	Molar gel composition (T = Si+Ge)				Material
	H_2O/T	R/T	HF/T	Si/Ge	
1[a]	20	0.5	0	0.6:0.4	**TON**+**MFI**+Arg[c]
2[a]	20	1	0	0.6:0.4	**MFI**+ε?[d]
3[a]	8	0.5	0.5	1:0	Amorphous
4[a]	8	0.5	0.5	0.8:0.2	IM−16+ε?[d]
5[a]	8	0.5	0.5	0.6:0.4	IM−16+ε?[d]
6[a]	8	0.5	0.5	0.4:0.6	*Q*[c]+IM−16
7[a]	8	0.5	0.5	0.2:0.8	*Q*[c]
8[a]	8	0.6	0.4	0.8:0.2	IM−16+ε?[d]
9[a]	20	1	0.5	0.6:0.4	IM−16+ε?[d]
10[a]	3	0.3	0.3	0.8:0.2	IM−16+ε?[d]
11[a]	20	1	1	0.8:0.2	IM−16+ε?[d]
12[a]	20	1	1	0.6:0.4	IM−16+ε?[d]
13[a]	20	1	1	0.5:0.5	IM−16+*Q*[e]
14[b]	20	1	1	0.8:0.2	IM−16+**MFI**
15[b]	20	1	1	0.6:0.4	IM−16+ε?[d]
16[b]	20	1	1	0.5:0.5	IM−16

Silica sources:
[a] TEOS (tetraethylorthosilicate).
[b] Aerosil 200.
[c] Argutite.
[d] ε?: small quantity of one or more unknown impurities.
[e] Quartz.

Information in table captions
Resolution of in-document symbols
Numerical reasoning for normalization
Heavy use of table footnotes
Sometimes indexed

Figure 3: Example table from the dataset, reproduced from Lorgouilloux et al. (2009, Table 1). This table is annotated with several of the challenges with table extraction in this dataset, including: (1) Generic table layout understanding; (2) Processing information related to tables, such as captions and footnotes; (3) Understanding and resolving in-document substitutions; and (4) Numerical reasoning to normalize ratios.

Table Understanding. One of the core challenge of this task is processing tables in the variety of forms in which they occur. Tables expressing synthesis parameters and recipes are difficult to construct: Experiments often involve the systematic variation of several different parameters, leading to a challenge in how to represent hierarchical data in many dimensions in an ultimately two-dimensional table. This results in a number of different formats. Figure 3 demonstrates perhaps the most common format, normalized rows per-experiment, but hierarchical representations that involve leaving cells blank to indicate a hierarchical grouping of experiments are also common, and pose a challenge for table understanding models.

Related Information. Information from text can be found in any section of a paper, and information necessary to understanding a table is frequently presented outside the table, whether in text, captions, footnotes, or otherwise. Figure 3 specifies the OSDA compound in the caption, and additionally specifies the expansion of several acronyms in table footnotes, which are placed directly below the table. Further, many papers introduce information necessary for table understanding in the text surrounding the tables. We note that table captions can be an edge case for some approaches to understanding document layout: The VILA (Shen et al., 2022a) model, for example, detects the table captions and footnotes as part of the table, which can lead some table understanding models to parse table captions and footnotes as further rows of the table, rather than footnotes. Further, understanding non-table information here requires the resolution of superscripts to their corresponding footnotes.

Numerical Reasoning. Synthesis procedures for zeolites are commonly expressed in terms of molar ratios of the components, and the choice of which element to which to normalize changes interpretation of numerical values in the table. For example, in Figure 3, ratios are scaled to the combination of silicon and germanium in the sample. By contrast, several other papers (and the final dataset) scale to only the quantity of silicon, requiring a normalization step that introduces a multiplicative factor to enable direct comparison of results across papers.

Within-document Reference Resolution. Before normalization can occur, tables often require the resolutions of symbols that are defined elsewhere in the document. In this case, the table headers indicate that the H_2O, R, and HF columns are normalized to T, which the upper header declares as the combination of silicon and germanium in the sample.

Sparsity. In many cases in the extracted dataset, columns will have values of 0, because a given element was not used. Systems that attempt this benchmark must not hallucinate non-zero values even when given a comprehensive schema of all items that may or may not be present.

3.4 Evaluation

Given the information extraction-based nature of this task, we consider an F1 metric, with some allowance for what is counted as a match. In the case of numeric columns, to allow for imprecision in normalization of ratios, we consider a “correct" answer to be within 0.1 of the true answer. In the case of the OSDA name column and the extracted products column, we expect an exact match on a lowercased version of the string with all punctuation replaced by an underscore; in the case of the extracted products column, we note that often, several products of a reaction are mentioned; we intend to improve the granularity of our evaluation in ongoing work. Given the large variance of the number of rows/data points extracted from individual articles, we consider a micro-averaged F1 score to be an appropriate choice.

For evaluation, we provide code that accepts a spreadsheet with the same header row as the original dataset (omitting the location rows), and is configurable per-dataset for which columns are to be evaluated. `None` values in predictions indicate that the model is not providing a response, to disambiguate from cases where the correct extraction is zero, or another common placeholder value. For our F1 metric, we consider any data that is available on the page (i.e. not annotated as being "not present") a candidate for extraction. A true positive is any data point that is available to the model and correctly extracted; false negatives are any point that the model fails to extract. False positives include both incorrectly extracted values and values that are not available to the model, but that it provided a value for anyway. True negatives are information not available to the model that it successfully does not provide a value for. We micro-average the F1 across papers, and additionally provide per-location F1 scores to indicate what sources of information models are adept at working with.

However, evaluation does pose additional challenges: While some tables translate straightforwardly between rows in the original table and the dataset, others are structured differently, using hierarchical indices, such that blank cells' content must be inferred, or tables with multiple levels of hierarchy, where one cell and the headers that index it correspond to a row in the final dataset. We call these tables *cross-indexed*. Further, while the table reproduced in Figure 3 uses identifiers for individual samples, that is not common in our dataset. As a result, there is no *a priori* alignment between rows in the dataset and rows produced from models solving this task.

To address this, we use a simple heuristic algorithm that attempts to align rows in the dataset with rows produced from the systems under evaluation, with strong priors towards the initial alignment being correct. Our algorithm begins by computing a row-wise score between all rows in the dataset and predictions. This score computes the match discussed above on all columns where information is within the provided context window, to avoid spurious matches on absent information. We then iterate through each row of the dataset, and choose the highest scoring predicted row to align with each row in the dataset. In the case of a score tie, the sequentially following row is assigned. Because of the varying structures of tables in the dataset, we additionally implement fallbacks in the case of a mismatched number of rows between the dataset and predictions. In the case where the model produces more rows than are observed in the dataset, each additional row is penalized as being false positives; in the case where the model produces too few rows, we construct placeholder rows of no predictions to indicate that the model has not provided an answer. We note both that this alignment strategy is not guaranteed to produce the optimal alignment, but also that any similar strategy will end up favoring models by potentially offering mistaken credit.

3.4.1 Permissive Evaluation

As a result of being focused on papers in a field where open-access publishing is not common, our dataset relies on users to reconstruct the dataset. This in turn relies on their access to the same journals we used when constructing the dataset, which we cannot reasonably assume. As a result, we provide a script to evaluate predictions made on a subset of the papers present in our dataset, rather than the whole dataset. In order that metrics computed in this way be comparable to the baselines, we additionally provide a script to re-evaluate the baseline predictions (which we also provide in our repo) on the same subset of data to provide an apples-to-apples comparison on the subset of papers that a given user has access to.

4 Baselines

Despite recent work investigating Large Language Models (LLMs) as possible automated scientists (Lu et al., 2024; Si et al., 2024), to our knowledge LLMs have never been systematically evaluated on research processes such as precise multi-document review and synthesis.

As a baseline, we evaluated a prompt-based strategy with a variety of Large Language Models, when available comparing unimodal (text only) and multimodal (text + vision) LLMs of comparable sizes. The models used are described in Table 2.

Our goal is for the model to perform both the information extraction and table normalization jointly when provided with either an image (300 dpi PNG) of the PDF page, or the raw underlying XML of the document split into chunks. For the visual modality, a PDF document is separated into individual pages. The VLM is prompted with the page and a text prompt, returning a list of possible data rows discovered in the image. Rows are then aggregated for each PDF document. For the XML modality, the XML corresponding to a document is split using a sliding window with `window size`

			precision	recall	f1
Dataset	Model	Modality			
aluminum	claude	pdf	0.044	1.000	0.084
		xml	0.058	0.911	0.108
	gpt4o	pdf	0.002	0.983	0.005
		xml	0.002	0.884	0.005
	llama3.3-70	xml	0.001	0.786	0.001
	molmo	pdf	0.000	0.522	0.001
	qwen2.5VL-72	pdf	0.001	0.759	0.002
zeolite	claude	pdf	0.169	0.766	0.277
		xml	0.204	0.938	0.335
	gpt4o	pdf	0.027	0.728	0.052
		xml	0.013	0.717	0.026
	llama3.3-70	xml	0.012	0.592	0.023
	molmo	pdf	0.005	0.316	0.010
	qwen2.5VL-72	pdf	0.029	0.739	0.056

Table 1: Baseline results. We note that in all cases that did not accept a full PDF or XML document, precision is extremely low, indicating that naïve approaches produce significant noise.

Model	Size	Open?	Multimodal?
Molmo	7B	✓	✓
Llama3.3	70B	✓	X
Qwen2.5	72B	✓	✓
GPT-4o	?	X	✓
Claude-3.7	?	X	✓

Table 2: Statistics of models used for the baseline experiments.

of 10000 characters, and a `stride` of 5000 characters; this ensures that no data components are split at a window boundary. Each window is passed to the LLM, and extracted rows are aggregated per document. The combined size of XML and prompt text meant that smaller open-weight models had insufficient context size.

Prompts were constructed in collaboration between two authors of this paper: One, a graduate student in NLP; the other, a graduate student in materials science. This development process allowed us to leverage insights coming from either NLP or material science expertise. The constructors were provided with three randomly selected articles from the dataset to act as a guide while developing their prompts. We intentionally restricted the prompt constructors' access to the full set of papers so that information and edge cases from the test set would not influence prompt design. The full text of the prompts is included in the Appendix.

Additionally, we define a consistent JSON structure and enforce structured output on the models when available. Outputs were then post-processed so that column names aligned with the evaluation data, and units were equivalent aligned across rows (e.g., converting degrees Kelvin to Celsius).

We used multimodal models to process image inputs, and text-only models to process XML inputs. For GPT-4o and Claude, we tested both image and XML inputs. This resulted in a total of 7 experiments for each dataset.

5 Results and Discussion

We summarize our high-level results in Table 1, and present additional experiments on non-TDM downloadable PDFs in appendix E. Overall, while many of the models achieve impressive recall scores, the precision of all models is unusably low, with the best models only reaching the scores of 0.2, and with the precision for several models, especially in the aluminum dataset, being within a rounding error of 0. We note that models that accepted the whole paper had significantly higher scores, indicating that naive approaches to splitting documents and aggregating predictions are ultimately insufficient; to improve our baseline results, we anticipate needing better methods to aggregate recipes across different pages, or to rely on models with context lengths long enough to process whole documents.

We also note that the models achieved impres-

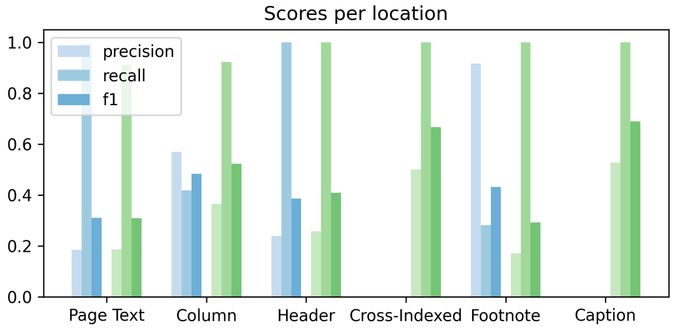

Figure 4: Location-based results. Blue bars indicate PDF results, green XML. Bars move from precision to recall to f1 from left to right. Cross-indexed tables and table captions are missing bars for the PDF modality because the PDFs we were able to scrape did not contain information presented in these ways. 3.

sive recall scores, especially in the aluminum alloy dataset. In annotating that dataset, we found that types of information tended to be expressed in highly formalized, similar ways. Properties were typically expressed in a table, while composition was expressed in page text. This indicates LLMs are highly structure dependent: where the information falls into a suitable structure, LLMs can be effective; this performance can rapidly degrade with variance in information presentation.

To substantiate this hypothesis, we plot the scores from our highest-performing model, Claude-3.7, against the location from which the data was extracted in the Zeolite dataset in Figure 4. Page text perhaps suffers the most from the precision issue, which could suggest that the degree of redundancy in page text is challenging for models to disambiguate; information in tables seems to achieve a better balance between precision and recall, though tables in the visual format do not achieve as high scores as in the XML format.

We also see differences across modalities. Footnotes seem better parsed in the visual medium, where information in table columns and headers is easier to parse from XML, where semantic markup might assist models' processing.

6 Related Work

Visually Rich Document Understanding The proposed benchmark bears many similarities to work in visually rich document understanding (VRDU). Tasks in VRDU, including to answer questions based on financial documents (Chen et al., 2022), or to understand forms (Jaume et al., 2019), receipts (Park et al.), or to perform information extraction on non-disclosure agreements and financial statements (Stanisławek et al., 2021). Each of these datasets emphasizes the use of visual document features as necessary information to understand the documents' contents. However, while many of these tasks focus on documents that have been scanned and had OCR applied to them, we focus in this paper on scientific documents that are natively digital. We note, however, that scientific literature from before digital typesetting remains of significant interest in many fields.

Scientific Document Understanding Separately from document understanding tasks more generally, work on understanding scientific documents is a growing field. Work like VILA (Shen et al., 2022b) implements document structure recognition on scientific publications, and DDD has historically relied on information extraction tools like ChemDataExtractor (Swain and Cole, 2016; Mavracic et al., 2021) and MatSciBERT (Gupta et al., 2022).

7 Conclusion

In this paper, we repurpose two tabular datasets for data-driven design in materials science as a benchmark for multimodal document understanding. We scope the problem to be whole-paper understanding tasks, in which models are expected to pull from a variety of information contexts to satisfy the goal, and expose two settings, multimodal and text-only. In evaluating a model on our benchmark, we see that while models attain high recall, potentially due to the redundancy of information across pages and chunks, their precision leaves much to be desired. We argue that benchmarks oriented towards data-driven design should be strong candidates on which to focus effort in information extraction, both to advance the state of the art in NLP, and for the utility to materials science.

Limitations

We see two major limitations on the scope of work in this paper. Firstly, we were not able to implement alternate paradigms of model prompting in our baselines. While the naive baseline does establish that a reasonable but not especially sophisticated method fails to produce the desired output, this is not a complete characterization of the current state of LM research.

Secondly, while we attempt to cover multiple domains, the range of materials science schemas varies more than we capture in our paper. In future work, we hope to extend this benchmark to further schemas.

References

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, and William Yang Wang. 2022. FinQA: A Dataset of Numerical Reasoning over Financial Data. *arXiv preprint*. ArXiv:2109.00122 [cs].

Callum J. Court and Jacqueline M. Cole. 2018. Auto-generated materials database of Curie and Néel temperatures via semi-supervised relationship extraction. *Scientific Data*, 5(1):180111. Publisher: Nature Publishing Group.

John Dagdelen, Alexander Dunn, Sanghoon Lee, Nicholas Walker, Andrew S. Rosen, Gerbrand Ceder, Kristin A. Persson, and Anubhav Jain. 2024. Structured information extraction from scientific text with large language models. *Nature Communications*, 15(1):1418. Publisher: Nature Publishing Group.

Tanishq Gupta, Mohd Zaki, NM Anoop Krishnan, and Mausam. 2022. Matscibert: A materials domain language model for text mining and information extraction. *npj Computational Materials*, 8(1):102.

Guillaume Jaume, Hazim Kemal Ekenel, and Jean-Philippe Thiran. 2019. FUNSD: A Dataset for Form Understanding in Noisy Scanned Documents. In *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, volume 2, pages 1–6.

Zach Jensen, Edward Kim, Soonhyoung Kwon, Terry Z. H. Gani, Yuriy Román-Leshkov, Manuel Moliner, Avelino Corma, and Elsa Olivetti. 2019. A Machine Learning Approach to Zeolite Synthesis Enabled by Automatic Literature Data Extraction. *ACS Central Science*, 5(5):892–899. Publisher: American Chemical Society.

Edward Kim, Kevin Huang, Alex Tomala, Sara Matthews, Emma Strubell, Adam Saunders, Andrew McCallum, and Elsa Olivetti. 2017. Machine-learned and codified synthesis parameters of oxide materials. *Scientific Data*, 4(1):170127. Publisher: Nature Publishing Group.

Yannick Lorgouilloux, Mathias Dodin, Jean-Louis Paillaud, Philippe Caullet, Laure Michelin, Ludovic Josien, Ovidiu Ersen, and Nicolas Bats. 2009. IM-16: A new microporous germanosilicate with a novel framework topology containing *d4r* and *mtw* composite building units. *Journal of Solid State Chemistry*, 182(3):622–629.

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. 2024. The ai scientist: Towards fully automated open-ended scientific discovery. *Preprint*, arXiv:2408.06292.

Juraj Mavracic, Callum J Court, Taketomo Isazawa, Stephen R Elliott, and Jacqueline M Cole. 2021. Chemdataextractor 2.0: Autopopulated ontologies for materials science. *Journal of Chemical Information and Modeling*, 61(9):4280–4289.

Santiago Miret and N. M. Anoop Krishnan. 2024. Are LLMs Ready for Real-World Materials Discovery? *arXiv preprint*. ArXiv:2402.05200 [cond-mat].

Elsa A. Olivetti, Jacqueline M. Cole, Edward Kim, Olga Kononova, Gerbrand Ceder, Thomas Yong-Jin Han, and Anna M. Hiszpanski. 2020. Data-driven materials research enabled by natural language processing and information extraction. *Applied Physics Reviews*, 7(4):041317.

Seunghyun Park, Seung Shin, Bado Lee, Junyeop Lee, Jaeheung Surh, Minjoon Seo, and Hwalsuk Lee. CORD: A Consolidated Receipt Dataset for Post-OCR Parsing.

Olivia P. Pfeiffer, Haihao Liu, Luca Montanelli, Marat I. Latypov, Fatih G. Sen, Vishwanath Hegadekatte, Elsa A. Olivetti, and Eric R. Homer. 2022. Aluminum alloy compositions and properties extracted from a corpus of scientific manuscripts and US patents. *Scientific Data*, 9(1):128. Publisher: Nature Publishing Group.

Zejiang Shen, Kyle Lo, Lucy Lu Wang, Bailey Kuehl, Daniel S. Weld, and Doug Downey. 2022a. VILA: Improving structured content extraction from scientific PDFs using visual layout groups. *Transactions of the Association for Computational Linguistics*, 10:376–392.

Zejiang Shen, Kyle Lo, Lucy Lu Wang, Bailey Kuehl, Daniel S Weld, and Doug Downey. 2022b. Vila: Improving structured content extraction from scientific pdfs using visual layout groups. *Transactions of the Association for Computational Linguistics*, 10:376–392.

Chenglei Si, Diyi Yang, and Tatsunori Hashimoto. 2024. Can llms generate novel research ideas? a large-scale human study with 100+ nlp researchers. *Preprint*, arXiv:2409.04109.

Tomasz Stanisławek, Filip Graliński, Anna Wróblewska, Dawid Lipiński, Agnieszka Kaliska, Paulina Rosalska, Bartosz Topolski, and Przemysław Biecek. 2021. Kleister: Key Information Extraction Datasets Involving Long Documents with Complex Layouts. volume 12821, pages 564–579. ArXiv:2105.05796 [cs].

Matthew C. Swain and Jacqueline M. Cole. 2016. ChemDataExtractor: A Toolkit for Automated Extraction of Chemical Information from the Scientific Literature. *Journal of Chemical Information and Modeling*, 56(10):1894–1904. Publisher: American Chemical Society.

Hengrui Zhang, Alexandru B. Georgescu, Suraj Yerramilli, Christopher Karpovich, Daniel W. Apley, Elsa A. Olivetti, James M. Rondinelli, and Wei Chen. 2024. Emerging Microelectronic Materials by Design: Navigating Combinatorial Design Space with Scarce and Dispersed Data. *arXiv preprint*. ArXiv:2412.17283 [cond-mat].

Zhiling Zheng, Oufan Zhang, Christian Borgs, Jennifer T. Chayes, and Omar M. Yaghi. 2023. ChatGPT Chemistry Assistant for Text Mining and the Prediction of MOF Synthesis. *Journal of the American Chemical Society*, 145(32):18048–18062. Publisher: American Chemical Society.

A Prompts

A.1 Aluminum XML

XML Document: {context} You are a materials science research assistant agent. Your task is to analyze papers from the materials science field and extract information about aluminium recipes. You have been given a document possibly describing the recipe. This will contain information about the recipe including: - The Aluminum Association (AA) alloy designation - Mechanical and/or thermal treatments - Weight percentage of all elements, including Cu, Mn, Si, Mg, Zn and more - Mechanical properties, including ultimate tensile strength (UTS), yield strength(YS), hardness and elogation

You must perform the following steps, using your own reasoning capabilities (which are significant and have been highly improved by Anthropic) and not relying on external tools. 1. Table CSV: Read the contents of the document or table, and duplicate that text or table within your response. Make sure to carefully read possible subscript (for instance, for element ratios in a molecular formula) from the text, and distinguish them from footnotes. Do not abbreviate the table. If values presented contain a range, calculate and write the mean of the range. 2. Extracted Text: Read beyond any tables to extract other recipe information that are not contained in the table text, but may be mentioned in the text of the paper or the caption of the table or image. 3. Property mapping: Identify which properties from the property list below are included in the document. Write out a mapping between the form within the text, and properties. 4. Abbreviations: Expand any abbreviations used in the recipe. For instance, if the recipe describes U = Na + Cl, that means that "U" represents the total amount of Na and Cl in the recipe. Write the expanded abbreviations and other relevant information. 5. Weight Percentage: Determine the weight percentage for each element excluding Al. Sometimes the text or table will already do this; in that case, replicate it from the document. But If the document has in atomic percentage, multiply the corresponding molar weights and calculate the weight percentage. You can write out the mathematical expressions used to perform these calculations. Always convert ratios and fractions to decimal form. 6. Rewrite the csv table or document text as a JSON containing adjusted values. For instance, if "U" (= Na + Cl) had its own column, create a column for Na and a column for Cl. The resulting recipe list must be a list of JSON objects, with each object corresponding to one recipe and its properties.

Ignore information related to the nature of the resulting compound, only focus on the parameters and instructions used to perform the recipe. If an expected value in the recipe (listed below) is missing, fill that value with the empty string.

The properties of interest are: - "AA" (str, the Aluminum Association (AA) alloy designation) - "temper" (str, mechanical and/or thermal treatment abbreviations mentioned in the document) - "YS [MPa]" (float, yield strength in MPa) - "UTS [MPa]" (float, ultimate tensile strength in MPa) - "elong [- "Hardness" (float, hardness in the unit of the document) - "Hardness UNIT" (str, the unit of hardness in the document) - "Has composition" (selection, True if the recipe has a spcified composition, nominal if it refers to a nominal composition by a standard (e.g. AA 6060), otherwise False) - Cu, Mn, Si, Mg, Zn, Cr, Fe, Ti, Zr, Ag, Be, Bi, C, Ca, Ce, Cd, Er, Ga, Ge, Hf, La, Li, Ni, P, Pb, Sc, Sn, Sr, V, Yb (float, label them one by one like above if exist in this selection of elements, otherwise do not include them in the JSON) Do not include any properties except for these (or properties which are equivalent)!

The JSON response should be in this format:

{{ "table csv": <open text>, "extracted text" <open text>, "property mapping": <open text>, "abbreviations": <open text>, "weight percentage": <open text>, "recipes": [{recipe_format}, {recipe_format}, ...,] }}

Sometimes there will not be valid information; in that case return the default value or an empty for the field. The output should only contain the JSON object without any additional text. Read the XML and begin the response below:

A.2 Aluminum PDF/Images

You are a materials science research assistant agent. Your task is to analyze papers from the materials science field and extract information about aluminium recipes. You have been given an PDF document possibly describing the recipe. This will contain information about the recipe including: - The Aluminum Association (AA) alloy designation - Mechanical and/or thermal treatments - Weight percentage of all elements, including Cu, Mn, Si, Mg, Zn and more - Mechanical properties, including ultimate tensile strength (UTS), yield strength(YS), hardness and elogation

You must perform the following steps, using your own reasoning capabilities (which are significant and have been highly improved by Anthropic) and not relying on external tools. 1. Table CSV: Read the contents of the document or table, and duplicate that text or table within your response. Make sure to carefully read possible subscript (for instance, for element ratios in a molecular formula) from the text, and distinguish them from footnotes. Do not abbreviate the table. If values presented contain a range, calculate and write the mean of the range. 2. Extracted Text: Read beyond any tables to extract other recipe information that are not contained in the table text, but may be mentioned in the text of the paper or the caption of the table or image. 3. Property mapping: Identify which properties from the property list below are included in the document. Write out a mapping between the form within the text, and properties. 4. Abbreviations: Expand any abbreviations used in the recipe. For instance, if the recipe describes U = Na + Cl, that means that "U" represents the total amount of Na and Cl in the recipe. Write the expanded abbreviations and other relevant information. 5. Weight Percentage: Determine the weight percentage for each element excluding Al. Sometimes the text or table will already do this; in that case, replicate it from the document. But If the document has in atomic percentage, multiply the corresponding molar weights and calculate the weight percentage. You can write out the mathematical expressions used to perform these calculations. Always convert ratios and fractions to decimal form. 6. Rewrite the csv table or document text as a JSON containing adjusted values. For instance, if "U" (= Na + Cl) had its own column, create a column for Na and a column for Cl. The resulting recipe list must be a list of JSON objects, with each object corresponding to one recipe and its properties.

Ignore information related to the nature of the resulting compound, only focus on the parameters and instructions used to perform the recipe. If an expected value in the recipe (listed below) is missing, fill that value with the empty string.

The properties of interest are: - "AA" (str, the Aluminum Association (AA) alloy designation) - "temper" (str, mechanical and/or thermal treatment abbreviations mentioned in the document) - "YS [MPa]" (float, yield strength in MPa) - "UTS [MPa]" (float, ultimate tensile strength in MPa) - "elong [- "Hardness" (float, hardness in the unit of the document) - "Hardness UNIT" (str, the unit of hardness in the document) - "Has composition" (selection, True if the recipe has a spcified composition, nominal if it refers to a nominal composition by a standard (e.g. AA 6060), otherwise False) - Cu, Mn, Si, Mg, Zn, Cr, Fe, Ti, Zr, Ag, Be, Bi, C, Ca, Ce, Cd, Er, Ga, Ge, Hf, La, Li, Ni, P, Pb, Sc, Sn, Sr, V, Yb (float, label them one by one like above if exist in this selection of elements, otherwise do not include them in the JSON) Do not include any properties except for these (or properties which are equivalent)!

The JSON response should be in this format:

{{ "table csv": <open text>, "extracted text" <open text>, "property mapping": <open text>, "abbreviations": <open text>, "weight percentage": <open text>, "recipes": [{recipe_format}, {recipe_format}, ...,] }}

Sometimes there will not be valid information; in that case return the default value or an empty for the field. The output should only contain the JSON object without any additional text. Read the PDF and begin the response below:

A.3 Zeolite XML

XML Document:

{context}

You are a materials science research assistant agent. Your task is to analyze papers from the materials science field and extract information about {recipe_type} recipes. You have been given an

XML document possibly describing the synthesis recipe. This will contain information about the recipe including: - Ratios of the reaction reagents, including {reagents} and other elements - Information on the temperature and duration of the reaction - The structure directing agent, which guides the formation of the zeolites

You must perform the following steps, using your own reasoning capabilities (which are significant and have been highly improved by {model_makers}) and not relying on external tools. 1. Read the contents of the document or table, and duplicate that text or table within your response. Make sure to carefully read possible subscript (for instance, for element ratios in a molecular formula) from the text, and distinguish them from footnotes. Do not abbreviate the table. If values presented contain a range, calculate and write the mean of the range. (table csv) 2. Read beyond any tables to extract other recipe information that are not contained in the table text, but may be mentioned in the text of the paper or the caption of the table. (extracted text) 3. Identify which properties from the property list below are included in the document. Many of the properties relate to quantities or ratios of reagents. Sometimes a table column will be named based on the source material (ie SiO2 for Si). Treat those as the corresponding element. Write out a mapping between the form within the text, and properties. (property mapping) 4. Expand any abbreviations used in the recipe. For instance, if the recipe describes U = Na + Cl, that means that "U" represents the total amount of Na and Cl in the recipe. Write the expanded abbreviations and other relevant information. 5. Determine the ratio for each reagent. Setting Silicon to "1", determine the proportion for each reagent relative to the silicon. Sometimes the text or table will already do this; in that case, replicate it from the document. But if a Si/Ge ratio of .5 is described, Si = 1 and Ge = 2. You can write out the mathematical expressions used to perform these calculations. Always convert ratios and fractions to decimal form. 6. Rewrite the csv table or document text as a JSON containing adjusted values. For instance, if "U" (= Na + Cl) had its own column, create a column for Na and a column for Cl. The resulting recipe list must be a list of JSON objects, with each object corresponding to one recipe and its properties.

Ignore information related to the nature of the resulting compound, only focus on the parameters and instructions used to perform the recipe. If an expected value in the recipe (listed below) is missing, fill that value with the empty string.

The properties of interest are: {properties} Do not include any properties except for these (or properties which are equivalent)!

The JSON response should be in this format:

{{ "table csv": <open text>, "extracted text" <open text>, "property mapping": <open text>, "formula abbreviations": <open text>, "ratio calculations": <open text>, "recipes": [{recipe_format}, {recipe_format}, ...,] }}

Sometimes there will not be valid information; in that case return an empty for the field. Read the XML and begin the response below:

A.4 Zeolite PDF/Images

You are a materials science research assistant agent. Your task is to analyze papers from the materials science field and extract information about {recipe_type} recipes. You have been given an image from a document possibly describing a synthesis recipe. This will contain information about the recipe including: - Ratios of the reaction reagents, including {reagents} and other elements - Information on the temperature and duration of the reaction - The structure directing agent, which guides the formation of the zeolites

You must perform the following steps, using your own reasoning capabilities (which are significant and have been highly improved by {model_makers}). 1. Read the contents of the document, including tables, and transcribe that text or table within your response. Make sure to carefully read possible subscript (for instance, for element ratios in a molecular formula) from the text, and distinguish them from footnotes. Do not abbreviate the table. If values presented contain a range, calculate and write the mean of the range. (table csv) 2. Read beyond any tables to extract other recipe information that are not contained in the table text, but may be mentioned in the text of the paper or the caption of the table. (extracted text) 3. Identify which properties from the property list below are included in the document. Many of the properties relate to quantities or ratios of reagents. Sometimes a table column will be named based on the source material (ie SiO2 for Si). Treat those as the corresponding element. Write out a mapping between the form within the text, and properties. (property mapping) 4. Expand any abbreviations used in the recipe. For instance, if the recipe describes U = Na + Cl, that means that "U" represents the total amount of Na and Cl in the recipe. Write the expanded abbreviations and other relevant information. 5. Determine the ratio for each reagent. Setting Silicon to "1", determine the proportion for each reagent relative to the silicon. Sometimes the text or table will already do this; in that case, replicate it from the document. But if a Si/Ge ratio of .5 is described, Si = 1 and Ge = 2. You can write out the mathematical expressions used to perform these calculations. Always convert ratios and fractions to decimal form. 6. Rewrite the csv table or document text as a JSON containing adjusted values. For instance, if "U" (= Na + Cl) had its own column, create a column for Na and a column for Cl. The resulting recipe list must be a list of JSON objects, with each object corresponding to one recipe and its properties.

Ignore information related to the nature of the resulting compound, only focus on the parameters and instructions used to perform the recipe. If an expected value in the recipe (listed below) is missing, fill that value with the empty string.

The properties of interest are: {properties} Do not include any properties except for these (or properties which are equivalent)!

The JSON response should be in this format:

{{ "table csv": <open text>, "extracted text" <open text>, "property mapping": <open text>, "formula abbreviations": <open text>, "ratio calculations": <open text>, "recipes": [{recipe_format}, {recipe_format}, ...,] }}

Sometimes there will not be valid information; in that case return an empty for the field. Response:

B Dataset details: zeolites

In this section, we provide a more detailed accounting of the zeolite dataset and the columns contained in it. Zeolites are produced by combining reaction components into a gel, which is then heat treated to grow crystals. Typically, the reaction components include precusor materials that act sources of silicon, germanium, and other elements, sources of OH^- ions, an acid, water, and an organic structure directing agent, or OSDA, which encourages crystal growth in specific ways. These make up the columns of the dataset, which we individually describe in table 3.

C Dataset details: aluminum alloys

In this section, we provide a more detailed accounting of the aluminum alloy dataset and the columns contained in it. This aluminum alloy dataset describes the physical characteristics of aluminum alloys relative to their composition. In table 4, we describe the groups of columns of the dataset.

D Worked Zeolite Example

Figure 3 represents indices 375-390 from our dataset. We reproduce the first four rows of this table here, and demonstrate how to extract the relevant columns in the first row. For easy comparison, we additionally present an un-annotated version of 3 here as 5.

If present, the silicon content is always the basis of normalization, and so receives a value of 1 in the `Si` column. This therefore leads us to normalize the germanium value, in the ratio of Si:Ge 0.4:0.6, to 0.667. This paper uses neither aluminum nor boron, leading to 0 values for both of those. Water and HF content are similarly normalized by dividing by 0.6.

In the table in Figure 3, the R column is interpreted as the OSDA, even though this is not specified in the paper. This is a common substitution, alongside others, such as using “T” as the basis for normalization. We therefore use the values in the R column for the SDA value.

Text found elsewhere on the page provides additional information that must be incorporated. Synthesis paragraph 2.1 implies that the OSDA is also the source of OH^- ions: “and 3-ethyl-1-methyl-3H-imidazol-1-ium bromide (98%, Solvionic), which was transformed into its OH^- form by ion exchange in water.” The time and temperature (170°C for 14 days) are from the same paragraph; 14 days must be normalized to 336 hours.

The name of the OSDA is specified in the table caption. The names of the products are extracted into column S, but must be expanded using the table footnotes to indicate that “Arg” is argutite, and “Q” is quartz.

This table demonstrates several of the challenges in this dataset, from table understanding, to resolving in-table references, having conventional knowledge, and using contextual text that is not explicitly part of the table being considered or extracted.

Dataset Columns	**Description**
Si	The molar fraction of silicon in the reaction gel. This is the basis for normalization, and is always 1 when silicon is present. If not, the gel is normalized to the quantity of germanium. This quantity is numeric.
Ge, Al, OH, H2O, HF, SDA, B	The molar fraction of each of these components. This molar fraction is calculated based on the precursor materials that are a source for those ions; using e.g. Al_2O_3, for instance, results in twice the quantity of Al^{3+} ions as the quantity of powder used. These quantities are numeric.
Time	The time, in hours, that the gel is processed for. This is usually expressed in days, and needs to be normalized to hours.
Temp	The temperature, in Celsius, that the gel is processed at. This is typically expressed in papers in either Celsius or Kelvin.
SDA Type	The name of the SDA used in the production process. This is typically given as a chemical name, and sometimes an abbreviation.
Extracted	The products extracted from the reaction. Zeolites are typically described by a three-letter code and number provided by the International Zeolite Association (IZA). Where zeolites are not produced in a reaction, the product is usually described as "amorphous".

Table 3: Descriptions of the columns in the zeolite dataset

Dataset Column	Description
AA	The named aluminum alloy series being used in this paper. These series are defined by the Aluminum Association (AA), and are typically a 4-number designations. See e.g. \url{https://www.aluminum.org/industry-standards}
Temper	The tempering process used on the aluminum alloy. This is typically expressed as a suffix of -T<number>on the series designation.
YS [MPa]	The yield strength of the alloy, measured in megapascals (MPa).
UTS [MPa]	The ultimate tensile strength of the alloy, measured in megapascals (MPa).
elong [%]	The degree that the alloy will elongate before fracturing.
Hardness	The measured hardness of the alloy.
Hardness UNIT	The unit in which hardness is measured.
Has comp [True / False / Nominal]	Whether the composition is measured in the paper's experiments (indicated by TRUE), is assumed based on the starting alloy (nominal), or entirely absent (FALSE)
Element columns (Cu, Mn, etc.)	The percentage weight of the given element measured. These will sum up to less than 100; the remainder is aluminum.

Table 4: Description of columns in the aluminum dataset.

Si	Ge	Al	OH	H_2O	HF	SDA	B	Time	Temp	SDA Type	Extracted
1	0.667	0	0.8335	33.34	0	0.8335	0	336	170	3-ethyl-1-meth...	TON+MFI+argutite
1	0.667	0	1.667	33.34	0	1.667	0	336	170	3-ethyl-1-meth...	MFI+unknown
1	0	0	0.5	8	0.5	0.5	0	336	170	3-ethyl-1-meth...	Amorphous
1	0.25	0	0.625	10	0.625	0.625	0	336	170	3-ethyl-1-meth...	IM-16+unknown

Table 5: Sample rows from our dataset, filtered from Jensen et al. (2019). This table represents the first four rows of the table seen in Figure 3. For space, we omit the columns where we describe where the data was located.

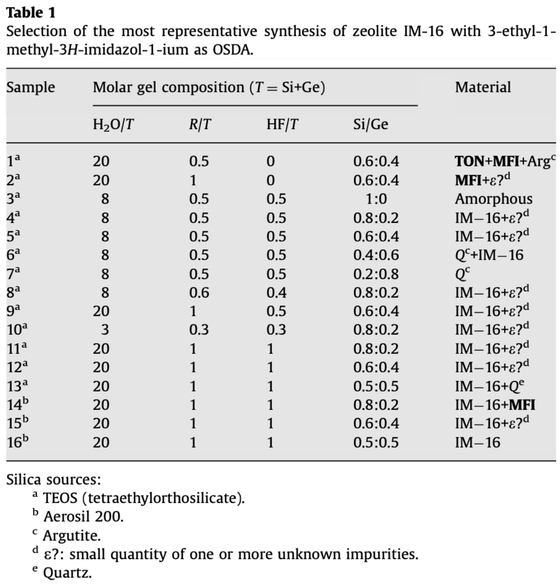

Table 1

Selection of the most representative synthesis of zeolite IM-16 with 3-ethyl-1-methyl-3*H*-imidazol-1-ium as OSDA.

Sample	Molar gel composition (T = Si+Ge)				Material
	H_2O/T	R/T	HF/T	Si/Ge	
1[a]	20	0.5	0	0.6:0.4	**TON**+**MFI**+Arg[c]
2[a]	20	1	0	0.6:0.4	**MFI**+ε?[d]
3[a]	8	0.5	0.5	1:0	Amorphous
4[a]	8	0.5	0.5	0.8:0.2	IM−16+ε?[d]
5[a]	8	0.5	0.5	0.6:0.4	IM−16+ε?[d]
6[a]	8	0.5	0.5	0.4:0.6	*Q*[c]+IM−16
7[a]	8	0.5	0.5	0.2:0.8	*Q*[c]
8[a]	8	0.6	0.4	0.8:0.2	IM−16+ε?[d]
9[a]	20	1	0.5	0.6:0.4	IM−16+ε?[d]
10[a]	3	0.3	0.3	0.8:0.2	IM−16+ε?[d]
11[a]	20	1	1	0.8:0.2	IM−16+ε?[d]
12[a]	20	1	1	0.6:0.4	IM−16+ε?[d]
13[a]	20	1	1	0.5:0.5	IM−16+*Q*[e]
14[b]	20	1	1	0.8:0.2	IM−16+**MFI**
15[b]	20	1	1	0.6:0.4	IM−16+ε?[d]
16[b]	20	1	1	0.5:0.5	IM−16

Silica sources:

[a] TEOS (tetraethylorthosilicate).

[b] Aerosil 200.

[c] Argutite.

[d] ε?: small quantity of one or more unknown impurities.

[e] Quartz.

Figure 5: Example table from the dataset, reproduced from Lorgouilloux et al. (2009, Table 1).

E Extended PDF Dataset Results

		precision	recall	f1
dataset	model			
aluminum	gpt4o	0.001	0.131	0.002
	molmo	0.000	0.066	0.001
	qwen2.5VL-72	0.001	0.132	0.001
zeolite	gpt4o	0.003	0.045	0.005
	molmo	0.000	0.002	0.000
	qwen2.5VL-72	0.003	0.024	0.005