

# PubTables-1M: Towards comprehensive table extraction from unstructured documents

Brandon Smock      Rohith Pesala      Robin Abraham  
Microsoft  
Redmond, WA  
brsmock,ropesala,robin.abraham@microsoft.com

## Abstract

Recently, significant progress has been made applying machine learning to the problem of table structure inference and extraction from unstructured documents. However, one of the greatest challenges remains the creation of datasets with complete, unambiguous ground truth at scale. To address this, we develop a new, more comprehensive dataset for table extraction, called PubTables-1M. PubTables-1M contains nearly one million tables from scientific articles, supports multiple input modalities, and contains detailed header and location information for table structures, making it useful for a wide variety of modeling approaches. It also addresses a significant source of ground truth inconsistency observed in prior datasets called oversegmentation, a novel canonicalization procedure. We demonstrate that these improvements lead to a significant increase in training performance and a more reliable estimate of model performance at evaluation for table structure recognition. Further, we show that transformer-based object detection models trained on PubTables-1M produce excellent results for all three tasks of detection, structure recognition, and functional analysis without the need for any special customization for these tasks. Data and code will be released at <https://github.com/microsoft/table-transformer>.

## 1. Introduction

A table is a compact, structured representation for storing and communicating data in documents and other manners of presentation. In its presented form, however, a table, such as the one in Fig. 1, may not explicitly represent its logical structure. This is a significant problem as a large amount of data is communicated through documents, and the absence of structure information can impede this data's use.

Inferring a table's structure from its presentation and converting it to a structured form is known as table extraction (TE). TE entails three subtasks [5], which we illustrate in

Figure 1. An example presentation table whose underlying logical structure is missing and must be inferred.

Fig. 2: table detection (TD), which locates the table; table structure recognition (TSR), which recognizes a table's rows, columns, and cells; and functional analysis (FA), which recognizes a table's keys and values. TE is challenging for automated systems [8, 11, 16, 22] due to the wide variety of formats, styles, and layouts found in presented tables. Recently, there has been a shift in the research literature from traditional rule-based methods [3, 10, 17] for TE to data-driven methods based on deep learning (DL) [13, 16, 21]. The primary advantage of DL methods is that they can learn to be more robust to the wide variety of table presentation formats. However, manually annotating tables for TSR is a difficult and time-consuming process [6]. To overcome this, researchers have turned recently to crowd-sourcing to construct larger datasets [8, 21, 22]. These datasets are assembled from documents created by thousands of authors, where each table's structure and content is annotated in a markup format such as HTML, XML, or LaTeX.

While crowd-sourcing solves the problem of dataset size, repurposing annotations unintended for TE and automatically converting these to ground truth presents its own challenges with respect to the completeness, consistency, quality, and explicitness of information. For instance, markup does not encode spatial coordinates for cells and encodes logical relationships implicitly through cues such as layout [19]. This lack of explicit information limits not only the range of potential modeling approaches, but also the quality control that can be done to verify the annotations' correctness.

Figure 2. Illustration of the three subtasks of table extraction addressed by the PubTables-1M dataset.

Another significant challenge for the use of crowd-sourced markup annotations is that their structures often exhibit an issue we refer to as **oversegmentation**. Oversegmentation occurs when a spanning cell in a header is incorrectly split into multiple grid cells. We illustrate this in Fig. 3. Oversegmentation in markup has no effect on how a presentation table appears when borders between cells are absent, leaving the implicit logical structure and interpretation unaffected. However, oversegmentation can lead to significant issues when used as ground truth for model training and evaluation.

The first issue is that an oversegmented annotation contradicts the logical interpretation of a table that its presentation is meant to suggest. For instance, oversegmenting a cell may indicate that its text applies to only one row when its presentation form suggests its text is meant to apply to several rows, as in the cell in column 1, row 3 in Fig. 3. This is problematic for use as ground truth to train a machine learning model to interpret a table's structure. Even if oversegmented annotations were considered a valid interpretation of a table's structure, allowing them leads to inconsistent ground truth, due to there then being multiple possible valid interpretations for a table's structure, as in Fig. 3. This violates the standard modeling assumption that there is exactly one correct ground truth for each table. Thus, datasets that contain oversegmented annotations lead to inconsistent, contradictory feedback during training and an underestimate of performance during evaluation.

To address these and other challenges, we develop a new large-scale dataset for table extraction called PubTables-1M. PubTables-1M contains nearly one million tables from scientific articles in the PubMed Central Open Access (PMCOA)

database. Among our contributions:

- PubTables-1M is nearly twice as large as the current largest comparable dataset and addresses all three tasks of table detection (TD), table structure recognition (TSR), and functional analysis (FA).
- Compared to prior datasets, PubTables-1M contains richer annotation information, including annotations for projected row headers and bounding boxes for all rows, columns, and cells, including blank cells. It also includes annotations on their original source documents, which supports multiple input modalities and enables a wide range of potential model architectures.
- We introduce a novel **canonicalization** procedure that corrects oversegmentation and whose goal is to ensure each table has a unique, unambiguous structure interpretation.
- To reduce additional sources of error, we implement several quality verification and control steps and provide measurable guarantees about the quality of the ground truth.
- We show that data improvements alone lead to a significant increase in performance for TSR, due both to improved training and a more reliable estimate of performance at evaluation.
- We apply the Detection Transformer (DETR) [1] for the first time to TD, TSR, and FA, and demonstrate how with PubTables-1M all three tasks can be addressed with a transformer-based object detection framework without any special customization for these tasks.

<sup>1</sup><https://www.ncbi.nlm.nih.gov/pmc/tools/openftlist/>

(a) Oversegmented structure annotation

(b) Canonical structure annotation

Figure 3. In this example, the structure annotation on the left is oversegmented, creating extra blank cells in the row and column headers. The canonical structure annotation on the right merges these cells and captures the table’s true logical structure. The blank cells in the top left corner are not part of the table and can be structured using any consistent scheme.

## 2. Related Work

**Structure recognition datasets** The first dataset to address all three TE tasks was the ICDAR-2013 dataset [5]. It remains popular for benchmarking due to its quality and relative completeness compared to other datasets. However, as a source of training data it is limited, containing only 257 tables for TD and TSR and 92 tables for FA.

Recently, larger datasets [2, 8, 21, 22] for TSR have been created by collecting crowd-sourced annotations automatically from existing documents. We summarize these in Tab. 1. The content and structure of each table is annotated in a markup format such as HTML. Various methods are used to determine each table’s spatial location within its containing document to create a correspondence between its markup and presentation. From there, datasets commonly frame the TSR task as: given an input table, output the structure—the assignment of cells to rows and columns—and text content for each cell, with image and HTML being example input and output formats, respectively, for these.

Most recently, two large datasets, FinTabNet and an enhanced version of PubTabNet, have added location information for cells, similar to ICDAR-2013. Adding location information enables the TSR task to be framed as outputting cell location instead of cell content, with cell content extraction being a trivial subsequent step. This increases the range of possible supervised modeling approaches. However, the bounding boxes for cells defined by these datasets cover only the text portion of each cell and exclude any additional whitespace that a cell might contain. This has a few implications, such as leaving bounding boxes for blank cells undefined and obscuring attributes of text contributed by whitespace, such as indentation and alignment. Therefore, one question left open by prior work is how to define bounding boxes for all cells, including blank cells.

There are additional challenges related to annotation completeness and quality that have not been addressed by prior datasets. In terms of completeness, prior large-scale datasets have also not included bounding boxes for rows and columns. Additionally, most datasets do not annotate the column header, and no large-scale dataset exists that specifies

the row header of a table. This not only limits the range of modeling approaches that can be applied to TSR but limits how completely the overall TE task can be solved.

Another open challenge is automated verification and measurement of annotation quality, which is important due to the impracticality of large-scale manual verification. Finally, prior datasets have not addressed the issue of oversegmented annotations. These are significant issues, as noise and mistakes in training data potentially harm learning, and in evaluation data potentially lead to an underestimate of model performance. Currently the extent to which these issues affect model training and evaluation is unexplored.

**Modeling approaches** One of the most common modeling approaches for TSR is to frame the task as some form of object detection [13, 16, 21]. Other approaches include those based on image-to-text [8] and graph-based approaches [2, 14]. While a number of general-purpose architectures, such as Faster R-CNN [15], exist for these modeling patterns, the unique characteristics of tables and the relative lack of training data have both contributed to the commonly observed underperformance of these models when applied to TSR out-of-the-box.

To get around deficiencies in training data, some approaches model TSR in ways that are only partial solutions to the task, such as row and column detection in DeepDeSRT [16], which ignores spanning cells, or image-to-text without text content, as in models trained on TableBank [8]. Other approaches use custom pipelines that branch into models to recognize tables with and without visible borders surrounding every cell [13, 21]. Many of the previously mentioned approaches also use engineered model components or custom training procedures, and incorporate rules or other unlearned processing stages tailored to the task, which brings in prior knowledge to lessen the burden placed on learning the task from data. Currently, no solution exists that uses a simple supervised learning approach with an off-the-shelf architecture, solves the TSR task completely, and achieves state-of-the-art performance.

Table 1. Comparison of crowd-sourced datasets for table structure recognition.

Dataset	Input Modality	# Tables	Cell Topology	Cell Content	Cell Location	Row & Column Location	Canonical Structure
TableBank [8]	Image	145K	X				
SciTSR [2]	PDF	15K	X	X			
PubTabNet [21, 22]	Image	510K <sup>z</sup>	X	X	X <sup>y</sup>		
FinTabNet [21]	PDF	113K	X	X	X <sup>y</sup>		
PubTables-1M (ours)	PDF	948K	X	X	X	X	X

Multiple input modalities, such as image or text, can be derived from annotated PDF data.

<sup>z</sup> The authors release annotations for 510K of the 568K total tables in their dataset.

<sup>y</sup> For these datasets, cell bounding boxes are given for non-blank cells only and exclude any non-text portion of a cell.

### 3. PubTables-1M

In this section, we describe the process used to develop PubTables-1M. To obtain a large source of annotated tables, we choose the PMCOA corpus, which consists of millions of public scientific articles. In the PMCOA corpus, each article is given in two forms: as a PDF document, which visually presents the article, and as an XML document, which provides a semantic description and hierarchical organization of the document's elements. Each table's content and structure is specified using standard HTML tags. However, because this data was not intended for use as ground truth for table extraction modeling, it does not explicitly label or guarantee multiple things that would be helpful for this purpose. For instance, although the same tables appear in both documents, no direct correspondence between them is given, like the location of each table. In terms of data quality, while tables are generally annotated reliably, it is not guaranteed that column headers are annotated completely or that text content as annotated exactly matches the text content as it appears in the PDF. Finally, some labels, such as the row header for each table, are not annotated at all.

The basic approach we take to overcome these issues is first we attempt to reliably infer as much missing annotation information as possible (for instance, the spatial location of each table) from the information that is present, then we verify that each annotation meets certain requirements for consistency. In some cases, we correct an annotation to attempt to make it more consistent, such as merging cells that are oversegmented. We consider certain requirements to be strict and samples whose annotations violate these are removed. This provides a set of conditions for quality and consistency that the annotations are guaranteed to meet. In the rest of this section, we describe these conditions and the steps we take to derive ground truth that meets them.

**Alignment** Text in a PDF document has location  $[x_{\min}; y_{\min}; x_{\max}; y_{\max}]$ , while text in an XML document appears inside semantically labeled tags. Because the correspondence between these is not given, the first step in

creating PubTables-1M is to match the text content from both. We process the PDF document into a sequence of characters each with their associated bounding box and use the Needleman-Wunsch algorithm [9] to align this with the character sequence for the text extracted from each table HTML. This connects the text within each HTML tag to its location with the PDF document. For each cell with text, we compute the text cell bounding box as the union of the bounding boxes for each character.

**Completion** Following alignment, we complete the spatial annotations to define bounding boxes for rows, columns, and the entire table. The bounding box for the table is defined as the union of all text cell bounding boxes. The  $x_{\min}$  and  $x_{\max}$  of the bounding box for each row are defined as the  $x_{\min}$  and  $x_{\max}$  of the table, giving every row the same horizontal length. The  $y_{\min}$  and  $y_{\max}$  of the bounding box for each row,  $m$ , are defined as the  $y_{\min}$  and  $y_{\max}$  of the union of the text cells for each cell whose starting row or ending row is  $m$ . Similarly, the  $y_{\min}$  and  $y_{\max}$  of the bounding box for each column are defined as the  $y_{\min}$  and  $y_{\max}$  of the table. The  $x_{\min}$  and  $x_{\max}$  of the bounding box for each column,  $n$ , are defined as the  $x_{\min}$  and  $x_{\max}$  of the union of the text cell for each cell whose starting column or ending column is  $n$ . From these definitions, the grid cell for each cell is defined as the union of the bounding boxes of the cell's rows intersected with the union of the bounding boxes for its columns. Unlike the text cell, the grid cell is defined even for blank cells.

**Canonicalization** The primary goal of the canonicalization step is to correct oversegmentation in a table's structure annotations. To do this we need to make assumptions about a table's intended structure. As the canonicalization algorithm itself is relatively simple, we first describe it, then detail the assumptions that motivate it. Put simply, canonicalization amounts to merging adjacent cells under certain conditions. The algorithm is given in Algorithm 1. But because it only operates on cells in the headers, HTML does not have a tag for a table's row header, and we observed that the column headers for tables in the PMCOA corpus are not always cor-



rect, we also include steps for inferring additional header row headers we only attempt to infer projected row headers cells that we believe can be reliably inferred in PMCOA (PRHs, also known as projected multi-level row headers [3], annotations. These additional steps significantly increase section headers [12], or super-rows [19]) and to infer cells the number of cells whose oversegmentation we are able to do that are in the first column of the header. The PRHs of a correct.

---

#### Algorithm 1 PubTables-1M Canonicalization

---

- 1: **ADD CELLS TO THE COLUMN AND ROW HEADERS**
  - 2:   Split every blank spanning cell into blank grid cells
  - 3:   if the first row starts with a blank cell then add the first row to the column header
  - 4:   if there is at least one row labeled as part of the column header then
    - 5:     while every column in the column header does not have at least one complete cell that only spans that column add the next row to the column header
  - 6:   end if
  - 7:   for each row do: if the row is not in the column header and has exactly one non-blank cell that occupies the first column then label it a projected row header
  - 8:   if any cell in the first column below the column header is a spanning cell or blank then add the column (below the column header) to the row header
  - 9: **MERGE CELLS**
  - 10:   for each cell in the column header do recursively merge the cell with any adjacent cells above and below in the column header that span the exact same columns
  - 11:   for each cell in the column header do recursively merge the cell with any adjacent blank cells below it if every adjacent cell below it is blank and in the column header
  - 12:   for each cell in the column header do recursively merge the cell with any adjacent blank cells above it if every adjacent cell above it is blank
  - 13:   for each projected row header do merge all of the cells in the row into a single cell
  - 14:   for each cell in the row header do recursively merge the cell with any adjacent blank cells below it
- 

We first assume that each table has an intended structure top-left corner. In this case, the blank cells are not part of the consistent with the Wang model [20], which in a study Wang table, so the assumptions about table structure do not suggest found was true for 97 percent of observed tables. Under how they should be grouped. We choose by convention to this model, the headers of the table each have a hierarchical merge all blank cells in the same column in a blank stub structure that corresponds logically to a tree. We assert that a head, which is consistent with the scheme in Line 10. for a structure annotation to be consistent with a table's logical structure, there should be exactly one cell for every tree node. We also assume that each value in the table is

indexed by a unique set of keys. We interpret this to mean Limitations While the stated goal of canonicalization is that each column in the body of the table corresponds to a applicable to any table structure annotation, we note that unique leaf node in the column header tree, and similarly that Algorithm 1 is designed to achieve this specifically for annotations in the PMCOA dataset. Canonicalizing tables from each row in the body corresponds to a unique leaf node in the row header tree (the index of a row or column can serve as another datasets may require additional assumptions and is key if necessary). These assumptions enable us to determine considered outside the scope of this work. Finally, it should if a row or column header is only partially annotated and if be noted that canonicalization does not guarantee mistake-so, to extend it to additional columns or rows, respectively. free annotations. Remaining issues are addressed using the However, to keep the precision of the algorithm high, for automated quality control procedure described next.

table can be identified using the rule in Line 7. Inference of the full row header is considered outside the scope of this work.

We also assume that any internal node in a header tree has at least two children. If not, ambiguity could arise in a table's logical structure because an internal node could optionally be split into a parent node and a single child node. The final assumptions we make are in regards to the root cause of oversegmentation in markup annotations. We assume that cells will only be oversegmented if an oversegmentation is consistent with the table's appearance. In practice, this means that cells with centered text will not be oversegmented in the direction of the alignment because this is likely to alter the table's appearance. For non-centered text, we expect that when cells in either header are oversegmented, this will happen vertically, as in Fig. 3b, and not horizontally due to the fact that text fills horizontal space before it fills vertical space, leaving more vertical space unused. Further, we expect that oversegmented cells in the row header will have text that is top aligned. Finally, we expect that when projected row headers are oversegmented, this will happen horizontally, not vertically, as a projected row header already occupies only one row.

Finally, there are two additional cases that we must handle by convention. One case is when one or more rows of blank grid cells are between a parent cell and all of its children cells in the column header. In this case, we can choose either to merge all of the blank cells with the parent cell above it or each with the child cell below it, and we choose the convention to merge all of the blank cells with the child, which occurs in Line 10. The final case is when a table has a blank stub head (according to the Wang model) in its

Table 2. Diversity and complexity of table instances in datasets for table structure recognition.

Dataset	# Tables <sup>z</sup>	# Unique Cell Topologies	Avg. # Tables Per Topology	Avg. Rows Per Table	Avg. Columns Per Table	Avg. Spanning Cells Per Table
SciTSR	14,933	2,622	5.70	9.28	5.19	0.77
PubTabNet	502,887	121,649	4.13	14.05	5.39	2.24
FinTabNet	112,826	9,565	11.80	11.93	4.36	1.01
PubTables-1M	947,642	250,910	3.78	13.41	5.46	3.01

<sup>z</sup> The number of ground truth tables in the dataset that we were able to successfully read and process.

Table 3. Estimated measure of oversegmentation for projected row headers (PRHs) by dataset. As PRHs are only one type of cell that can be oversegmented, this is a partial survey of the total oversegmentation in these datasets.

Dataset	Total Tables Investigated <sup>y</sup>	Total Tables with a PRH	Tables with an Oversegmented PRH		
			Total	% (of Total with a PRH)	% (of Total Investigated)
SciTSR	10,431	342	54	15.79%	0.52%
PubTabNet	422,491	100,159	58,747	58.65%	13.90%
FinTabNet	70,028	25,637	25,348	98.87%	36.20%
PubTables-1M	761,262	153,705	0	0%	0%

<sup>y</sup> We exclude tables with fewer than five rows; to avoid column header rows we skip the first four rows when searching for PRHs.

PRH = projected row header; these can be reliably detected in datasets without any prior row or column header annotations.

**Quality control** Because PubTables-1M is too large to be Dataset statistics and splits In total, PubTables-1M contains 947,642 tables for TSR, of which 52.7% are complex. Iter these from the data. First, as tables rendered from (have at least one spanning cell). Prior to canonicalization, markup should not contain overlapping rows or overlapping 40.1% of the tables were originally annotated as complex. columns, we discard any table where this occurs, as these are Canonicalization adjusted the annotations in some way for likely due to errors in the source markup or introduced by the 34.7% of all tables, or 65.8% of complex tables. alignment process. Next, to ensure text annotation quality, In Tab. 2, we compare the diversity and complexity of we compare the edit distance between the non-whitespace PubTables-1M's samples to other datasets. To measure diversity, we count the number of unique table structure layouts (or cell topologies) in each dataset. As can be seen, PubTables-1M includes more diversity per sample, a wider range of table structures, and layouts that are more complex. We filter out any tables for which the normalized edit distance between these averaged over every cell is above 0.05. We do not force the text from each table to be equal, overall compared to prior datasets. as the PDF text can differ even when everything is annotated In Tab. 3, we attempt to measure and compare the amount of oversegmentation present in each dataset. Precisely measuring this requires annotations for the row and column headers. But because other datasets lack these, we instead measure just oversegmented projected row headers (PRHs), provide an unambiguous assignment of all words in the table which can be detected reliably without explicit annotations to cells, we also compute the average fraction of overlap using the rule in Line 7. To account for missing column header annotations, we do not start looking for PRHs until and its most overlapping grid cell, and discard tables with an at least the fifth row, which assumes the column header occupies at most four rows, and we simply exclude any tables in a table (defined in Sec. 4) and remove tables with more than 100 as outliers, which discards less than 0.1% of tables. footers, we also do not count any detected PRHs that are the PubTables-1M is the first dataset that verifies annotations last rows of the table. A detected PRH is oversegmented if at the cell level and provides a measurable assurance of its row contains a blank cell. As can be seen, a significant consistency for the ground truth. Thus improving the explicit amount of oversegmentation is present in crowd-sourced fitness of information is valuable in part because it leads to datasets that have not been canonicalized. While a dataset more opportunities for catching inconsistencies and errors with oversegmentation can be self-consistent, like FinTabNet, combining datasets with different annotation schemes

Table 4. Test performance of models on PubTables-1M using object detection metrics.

Task	Model	AP	AP <sub>50</sub>	AP <sub>75</sub>	AR
TD	Faster R-CNN	0.825	0.985	0.927	0.866
	DETR	0.970	0.995	0.989	0.985
TSR + FA	Faster R-CNN	0.722	0.815	0.785	0.762
	DETR	0.912	0.971	0.948	0.942

Figure 4. An example table with dilated bounding box annotations for different object classes for jointly modeling table structure recognition and functional analysis.

could exacerbate inconsistency or even lead to over fitting.

We split PubTables-1M randomly into train, validation, and test sets at the document level using an 80/10/10 split. For TSR and FA, this results in 758,849 tables for training; 94,959 for validation; and 93,834 for testing. For TD, there are 460,589 pages containing tables for training; 57,591 for validation; and 57,125 for testing. Note that these tables span one page only—tables that span multiple pages are considered outside the scope of this work.

#### 4. Proposed Model

We model all three tasks of TD, TSR, and FA as object detection with images as input. For TD, we use two object classes: table and table rotated. The table rotated class corresponds to tables that are rotated counterclockwise 90 degrees.

**TSR and FA model** We use a novel approach that models TSR and FA jointly using six object classes: table, table column, table row, table column header, table projected row header, and table spanning cell. We illustrate these classes in Fig. 4. The intersection of each pair of table column and table row objects can be considered to form a seventh implicit class, table grid cell. These objects model a table’s hierarchical structure through physical overlap.

For the TSR and FA model, we use bounding boxes that are dilated. To create dilated bounding boxes, for each pair of adjacent rows and each pair of adjacent columns, we expand their boundaries until they meet halfway, which fills the empty space in between them. Similarly we expand the objects from the other classes so their boundaries match the adjustments made to the rows and columns they occupy. After, there are no gaps or overlap between rows, between columns, or between cells.

To demonstrate the proposed dataset and the object detection modeling approach, we apply the Detection Transformer (DETR) [1] to all three TE tasks. We train one DETR model for TD and one DETR model for both TSR and FA. For comparison, we also train a Faster R-CNN [15] baseline model for the same tasks. All models use a ResNet-18 backbone pre-trained on ImageNet with the first few layers frozen. We avoid custom engineering the models and training procedures for each task, using default settings wherever possible to allow the data to drive the result.

#### 5. Experiments

In this section, we report the results of training the baseline models for all three tasks on data derived from PubTables-1M. We report the performance of the models for table detection in Tab. 4. For TD, DETR slightly outperforms Faster R-CNN on AP<sub>50</sub> but significantly outperforms on AP, demonstrating superior performance for table localization. This gap suggests that the dataset is not trivial to learn, yet is large and consistent enough for a model to learn effectively.

For TSR and FA, we train three baseline models: Faster R-CNN and DETR on the canonicalized data, and DETR on the original, non-canonical (NC) annotations (DETR-NC). We report the results using object detection metrics for models trained on canonical data in Tab. 4, which measures performance jointly on TSR and FA, and report results for all models using TSR-only metrics in Tab. 5. We evaluate DETR-NC on both canonical and non-canonical test data.

For assessing TSR performance, we report several metrics including the table content accuracy metric ( $Acc_{Con}$ ), which is the percentage of tables whose text content matches the ground truth exactly for every cell, the F-score of the directed adjacency relations ( $F_{AR_{Con}}$ ) metric [4], and the recently proposed GriTS [18] metrics for partial table correctness. GriTS assesses cell topology recognition ( $GriTS_{Top}$ ), cell content recognition ( $GriTS_{Con}$ ), and cell location recognition ( $GriTS_{Loc}$ ) using the same overall form,

$$GriTS_f(A; B) = \frac{2 \sum_{i,j} f(A_{ij}; B_{ij})}{|A|_j + |B|_j} \quad (1)$$

where  $A$  and  $B$  are the ground truth and predicted matrices of grid cells, respectively.

Table 5. Test performance of the TSR + FA models on PubTables-1M using TSR-specific metrics.

Test Data	Model	Table Category	Acc <sub>Con</sub>	GriTS <sub>Top</sub>	GriTS <sub>Con</sub>	GriTS <sub>Loc</sub>	DAR <sub>Con</sub>
Non-Canonical	DETR-NC	Simple	0.8678	0.9872	0.9859	0.9821	0.9801
		Complex	0.5360	0.9600	0.9618	0.9444	0.9505
		All	0.7336	0.9762	0.9761	0.9668	0.9681
Canonical	DETR-NC	Simple	0.9349	0.9933	0.9920	0.9900	0.9865
		Complex	0.2712	0.9257	0.9290	0.9044	0.9162
		All	0.5851	0.9576	0.9588	0.9449	0.9494
	Faster R-CNN	Simple	0.0867	0.8682	0.8571	0.6869	0.8024
		Complex	0.1193	0.8556	0.8507	0.7518	0.7734
		All	0.1039	0.8616	0.8538	0.7211	0.7871
	DETR	Simple	0.9468	0.9949	0.9938	0.9922	0.9893
		Complex	0.6944	0.9752	0.9763	0.9654	0.9667
		All	0.8138	0.9845	0.9846	0.9781	0.9774

We observe that DETR trained on the canonical data for all three table extraction tasks and showed for the first time produces strong results for TSR and FA, outperforming all time that it is possible to achieve state-of-the-art performance of the other baseline models. Comparing DETR-NC trained within a standard object detection framework without the use of canonicalized data and evaluated on NC ground truth versus DETR trained need for any special customization for these tasks. While we and evaluated on canonical ground truth, we observe that do not believe this work raises any issues regarding negative the use of canonicalized data improves performance across impacts to society, we welcome a discussion on any potential both simple and complex tables. This is even more apparent impacts raised by others. in exact match accuracy, where canonicalized data boosts performance from 0.5360 to 0.6944 for complex tables.

To consider the impact that canonicalization has on facilitating a more reliable evaluation, we compare DETR-NC evaluated on canonical versus NC data. Even though it is trained on NC data, accuracy for simple tables is much higher when DETR-NC is evaluated on canonical data (0.9349) than when it is evaluated on NC data (0.8678). This clearly demonstrates that the canonical data is less noisy and contributes to a more reliable evaluation.

Finally, for DETR-NC we observe a significant drop across all metrics on complex tables when the evaluation is changed from NC data to canonical data. Setting aside the amount of self-consistency present within each set of annotations, this gap further highlights how much the two sets are simply different. To the extent that canonical structure annotations are considered more useful, models trained on canonical data can also be considered more useful, in addition to being more consistent.

## 6. Conclusion

In this paper, we introduced PubTables-1M, a new dataset for table extraction in unstructured documents that addresses the challenge of creating complete, reliable ground truth at scale. We called attention to the problem that oversegmentation in markup annotations leads to ambiguous ground truth in crowd-sourced datasets, and proposed a canonicalization procedure to address this. We demonstrated that improvements to the ground truth have a significant positive impact on model performance. Finally, we adopted DETR

## 7. Future Work

In the future, we hope to expand the proposed methods and canonicalization from tables in scientific articles to additional domains such as financial documents. We also hope to address the open challenge of accurately annotating row headers in large-scale datasets, which will enable even more complete solutions for table extraction. Finally, table extraction is often just one stage in larger pipelines for document understanding and information retrieval, and developing end-to-end systems in these areas is an important direction with its own challenges. We hope that releasing a large pool of detailed annotations from the PMCOA corpus in particular can further progress in this area.

## 8. Acknowledgments

We would like to thank Pramod Sharma, Natalia Larios Delgado, Joseph N. Wilson, Mandar Dixit, John Corring, Ching Pui WAN, and the anonymous reviewers for helpful discussions and feedback while preparing this manuscript.

## References

- [1] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. *European Conference on Computer Vision* pages 213–229. Springer, 2020. 2, 7



- [2] Zewen Chi, Heyan Huang, Heng-Da Xu, Houjin Yu, Wanxuan Yin, and Xian-Ling Mao. Complicated table structure recognition. *arXiv preprint arXiv:1908.04729*, 2019. 3, 4
- [3] Wolfgang Gatterbauer, Paul Bohunsky, Marcus Herzog, Bernhard Krüpl, and Bernhard Pollak. Towards domain-independent information extraction from web tables. *Proceedings of the 16th international conference on World Wide Web* pages 71–80, 2007. 1
- [4] Max Gišbel, Tamir Hassan, Ermelinda Oro, and Giorgio Orsi. A methodology for evaluating algorithms for table understanding in PDF documents. *Proceedings of the 2012 ACM symposium on Document engineering* pages 45–48, 2012. 7
- [5] Max Gišbel, Tamir Hassan, Ermelinda Oro, and Giorgio Orsi. ICDAR 2013 table competition. *2013 12th International Conference on Document Analysis and Recognition* pages 1449–1453. IEEE, 2013. 1, 3
- [6] Jianying Hu, Ramanujan Kashi, Daniel Lopresti, George Nagy, and Gordon Wilfong. Why table ground-truthing is hard. In *Proceedings of Sixth International Conference on Document Analysis and Recognition* pages 129–133. IEEE, 2001. 1
- [7] Jianying Hu, Ramanujan S Kashi, Daniel P Lopresti, and Gordon Wilfong. Table structure recognition and its evaluation. In *Document Recognition and Retrieval VIII*, volume 4307, pages 44–55. International Society for Optics and Photonics, 2000. 5
- [8] Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, Ming Zhou, and Zhoujun Li. Tablebank: Table benchmark for image-based table detection and recognition. *Proceedings of The 12th Language Resources and Evaluation Conference* pages 1918–1925, 2020. 1, 3, 4
- [9] Saul B Needleman and Christian D Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology* 48(3):443–453, 1970. 4
- [10] Ermelinda Oro and Massimo Ruffolo. TREX: An approach for recognizing and extracting tables from PDF documents. In *2009 10th International Conference on Document Analysis and Recognition* pages 906–910. IEEE, 2009. 1
- [11] Shubham Singh Paliwal, D Vishwanath, Rohit Rahul, Monika Sharma, and Lovekesh Vig. Tablenet: Deep learning model for end-to-end table detection and tabular data extraction from scanned document images. *2019 International Conference on Document Analysis and Recognition (ICDAR)* pages 128–133. IEEE, 2019. 1
- [12] David Pinto, Andrew McCallum, Xing Wei, and W Bruce Croft. Table extraction using conditional random fields. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval* pages 235–242, 2003. 5
- [13] Devashish Prasad, Ayan Gadpal, Kshitij Kapadni, Manish Visave, and Kavita Sultanpure. CascadeTabNet: An approach for end to end table detection and structure recognition from image-based documents. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops* pages 572–573, 2020. 1, 3
- [14] Shah Rukh Qasim, Hassan Mahmood, and Faisal Shafait. Rethinking table recognition using graph neural networks. In *2019 International Conference on Document Analysis and Recognition (ICDAR)* pages 142–147. IEEE, 2019. 3
- [15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. *arXiv preprint arXiv:1506.01497*, 2015. 3, 7
- [16] Sebastian Schreiber, Stefan Agne, Ivo Wolf, Andreas Dengel, and Sheraz Ahmed. DeepDeSRT: Deep learning for detection and structure recognition of tables in document images. In *2017 14th IAPR international conference on document analysis and recognition (ICDAR)*, volume 1, pages 1162–1167. IEEE, 2017. 1, 3
- [17] Alexey O Shigarov. Table understanding using a rule engine. *Expert Systems with Applications* 42(2):929–937, 2015. 1
- [18] Brandon Smock, Rohith Pesala, and Robin Abraham. GriTS: Grid table similarity metric for table structure recognition. *arXiv preprint arXiv:2203.12555*, 2022. 7
- [19] Ashwin Tengli, Yiming Yang, and Nian Li Ma. Learning table extraction from examples. *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics* pages 987–993, 2004. 1, 5
- [20] Xinxin Wang. Tabular abstraction, editing, and formatting, 1996. 5
- [21] Xinyi Zheng, Douglas Burdick, Lucian Popa, Xu Zhong, and Nancy Xin Ru Wang. Global table extractor (GTE): A framework for joint table identification and cell structure recognition using visual context. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision* pages 697–706, 2021. 1, 3, 4
- [22] Xu Zhong, Elaheh ShafaeiBavani, and Antonio Jimeno Yepes. Image-based table recognition: data, model, and evaluation. *arXiv preprint arXiv:1911.10683*, 2019. 1, 3, 4