Visual Understanding of Complex Table Structures from Document Images

Sachin Raja
IIIT-Hyderabad
sachin.raja@research.iiit.ac.in

Ajoy Mondal
IIIT-Hyderabad
ajoy.mondal@iiit.ac.in

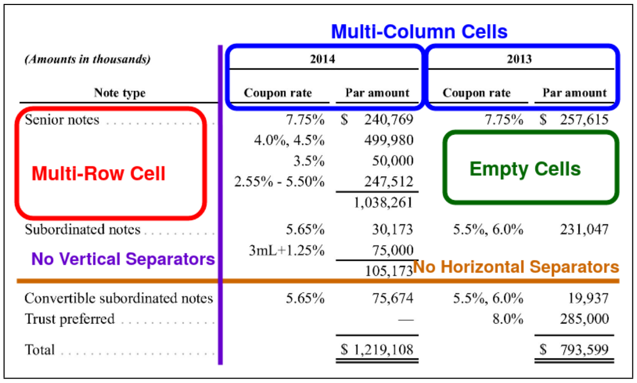
Jawahar C V
IIIT-Hyderabad
jawahar@iiit.ac.in

Abstract

Table structure recognition is necessary for a comprehensive understanding of documents. Tables in unstructured business documents are tough to parse due to the high diversity of layouts, varying alignments of contents, and the presence of empty cells. The problem is particularly difficult because of challenges in identifying individual cells using visual or linguistic contexts or both. Accurate detection of table cells (including empty cells) simplifies structure extraction and hence, it becomes the prime focus of our work. We propose a novel object-detection-based deep model that captures the inherent alignments of cells within tables and is fine-tuned for fast optimization. Despite accurate detection of cells, recognizing structures for dense tables may still be challenging because of difficulties in capturing long-range row/column dependencies in presence of multi-row/column spanning cells. Therefore, we also aim to improve structure recognition by deducing a novel rectilinear graph-based formulation. From a semantics perspective, we highlight the significance of empty cells in a table. To take these cells into account, we suggest an enhancement to a popular evaluation criterion. Finally, we introduce a modestly sized evaluation dataset with an annotation style inspired by human cognition to encourage new approaches to the problem. Our framework improves the previous state-of-the-art performance by a 2.7% average F1-score on benchmark datasets.

1. Introduction

A fine-grained understanding of complex document objects such as tables, charts, and graphs in document images is challenging. We focus on table structure recognition, which is a precursor to semantic table understanding. Table structure recognition generates a machine-interpretable output for a given table image, which encodes its layout according to a pre-defined standard [30, 17, 20, 42, 4, 39, 24]. Table structure recognition is difficult due to (a) inconsistency in size and density of tables, (b) absence of horizontal and/or vertical separator lines, (c) variation in table cells' shapes and sizes, (d) table cells spanning multiple rows and (or) columns, (e) presence of empty cells, and (f) cells with multi-line content [12, 36, 13, 9, 15, 31]. Figure 1 visually illustrates some of the challenges.


Multi-Column Cells
Multi-Row Cell
Empty Cells
No Vertical Separators
No Horizontal Separators

(Amounts in thousands)	2014		2013	
Note type	Coupon rate	Par amount	Coupon rate	Par amount
Senior notes	7.75%	$ 240,769	7.75%	$ 257,615
	4.0%, 4.5%	499,980		
	3.5%	50,000		
	2.55% - 5.50%	247,512		
		1,038,261		
Subordinated notes	5.65%	30,173	5.5%, 6.0%	231,047
	3mL+1.25%	75,000		
		105,173		
Convertible subordinated notes	5.65%	75,674	5.5%, 6.0%	19,937
Trust preferred		—	8.0%	285,000
Total		$ 1,219,108		$ 793,599

Figure 1. Demonstrates the challenges in table structure recognition task including absence of horizontal and vertical separators, multi-row/column spanning cells and empty cells.

Structure recognition of tables generally requires it to be broken down into cells first and then building associations between them. Cell detection is carried out using either visual or linguistic cues or both. As a precursor to obtain a good structure recognition performance, it is imperative to detect cells that are highly accurate and closely overlap with the ground truth. In few instances where access to machine-readable PDFs is available, it becomes easier to identify content and its location for every table cell. Detection of table cells as independent objects is challenging, as discussed earlier. Contrarily, since tables generally adhere to an inherent structural alignment, it is relatively easier to locate columns and rows. However, that would split cells that span multiple rows/columns. In this work, we locate table cells independently and through detection of rows and columns while preserving the multi-row and multi-column spanning structures. Our results demonstrate improved F1-scores for cell detection and better localization of empty cells.

This brings up an interesting thought: "How to interpret table cells without content and whether they carry any semantic meaning or not?". The absence of text in a table region may or may not suggest the presence of empty table cells, which are therefore difficult to detect. In most cases, cells that have no content might carry implicit semantic meanings. For example, an empty cell in a numeric column in balance sheets would either indicate a zero value or 'not applicable'. Similarly, the row header cell corresponding to the 'total' or 'sum' values is usually left blank. There might also be cases where an empty cell might span multiple rows and/or columns, such as a row header cell. In such instances, not correctly detecting empty cells would result in a loss of information during semantic parsing of the tables. Therefore, we emphasize the detection of empty cells and propose enhancing the existing vision-based criteria [7] to consider empty cells for evaluation.

The natural follow up question becomes: "What characterizes a good cell detection performance in a visual context?". In natural object detection, Intersection over Union (IoU) measure estimates of how well an object is detected. However, there are two concerning factors for cell detection: (i) How are the ground truth cell bounding-boxes annotated? (ii) What is the IoU threshold value used to compute evaluation metrics? For table cells, most datasets [22, 4, 6, 17, 42, 41] have cell box annotation that spans the smallest rectangle encapsulating its content. This annotation style misses on the bounding boxes for empty cells and on cells' inherent alignment constraints. Further, most cell detection methods [7, 41] evaluate using an Intersection over Union (IoU) threshold of 0.6, which might not always correspond to capturing the entire cell content. In light of these challenges, we believe it is important for a cell detection method to perform well on high IoU thresholds. In that regard, there also arises a need for a standard evaluation dataset. Its ground truth cell boxes preserve their native alignment constraints (just as we humans perceive tables) and have annotations for empty cells. We present Table Understanding for Complex Documents (TUCD) as an evaluation dataset consisting of 4500 manually annotated table images from business domain with a high diversity of table layouts having complex structures (samples shown in the supplementary material).

To detect table cells, we propose TOD-Net, where we augment the cell detection network of TabStruct-Net [24] with additional loss components to further improve the table object performance (rows/columns/cells) detection. These losses (formulated as regularizers) improve cell detection performance on high IoU thresholds by pairwise modelling of structural constraints. It allows for an improved bounding box detection despite presence of non-informative visual features in a specific table region using information from other cells detected in a different region of the table.

Once table cells are located precisely, extracting structure as an XML or any other predefined format is relatively easier. However, for extremely dense tables with many multi-row and multi-column spanning cells, it may still be challenging to build associations between cells that are far apart in the two-dimensional space. To handle this problem, we propose TSR-Net for structure recognition which uses the existing DGCNN architecture [22]. Our formulation uses rectilinear adjacencies instead of row/column adjacencies [22, 24]. Recursive parsing of rectilinear adjacencies helps to build better long-range visual row/column associations.

Our contributions can be summarized as follows:

- Introduce channel attention [19] for table object detection and define two additional regularizers — continuity and overlapping loss between every pair of cells in addition to the alignment loss from [24]. We use trainable loss-weights for these losses and formulate a min-max optimization problem for faster convergence.

- Formulate structure recognition using rectilinear adjacencies instead of row/column adjacencies, eliminating the need for complex post-processing heuristics for generating row and column spanning information for every cell.

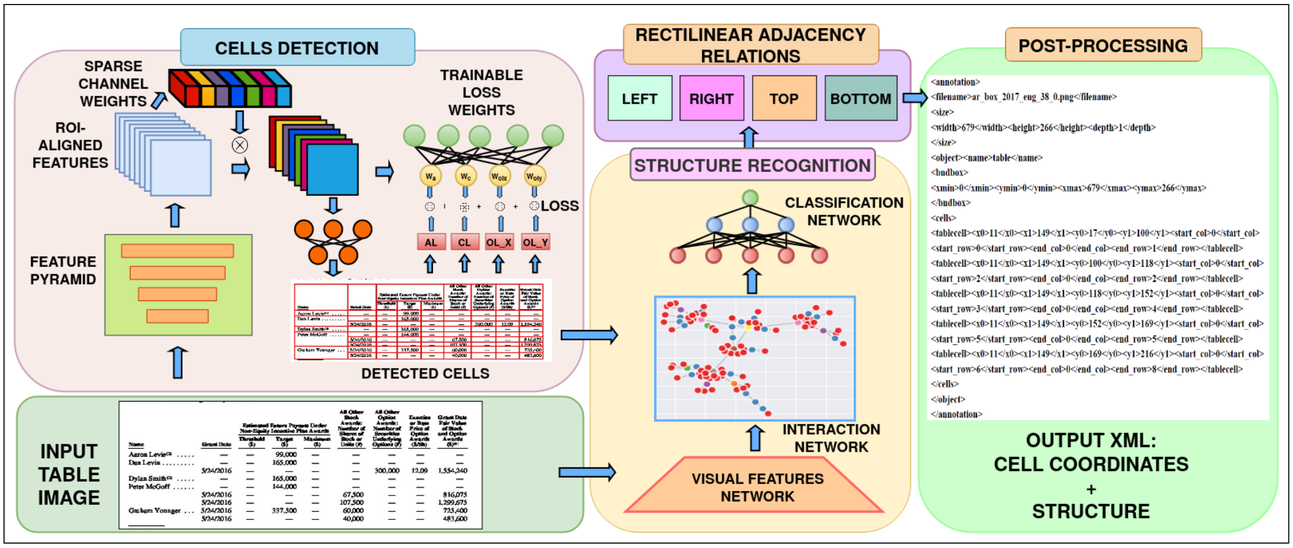
- Introduce modestly sized manually annotated TUCD as an evaluation dataset comprising 4500 table images from publicly available annual reports.

- Suggest improvements to the existing criterion proposed in [7] for a stricter evaluation of table structure recognition and demonstrate significantly improved performance on relatively higher IoU thresholds of 0.7 and above compared to the state-of-the-art methods.

- We demonstrate improved performance on cell detection through intermediate row and column detection tasks.

2. Related Work

Early methods [36, 12] on table structure recognition primarily depend on hand-crafted features and heuristics (horizontal and vertical ruling lines, spacing, and geometric analysis). However, these usually make strong assumptions about table layouts for a domain agnostic algorithm. Some recent data-driven methods include works by [1, 32, 21, 33]. Cognitive methods in this space broadly classified into five categories — image-to-sequence models [17, 2, 14], segmentation networks [26, 18, 20, 23], graph formulations [22, 4, 24], conditional generative adversarial networks [16] and a recent multi-modal method by [40]. A combination of heuristics and deep learning methods was also proposed [30] based on splitting the table into sub-cells, and then merging semantically connected



```
<annotation>
<filename>ar_box_2017_eng_38_0.png</filename>
<size>
<width>679</width><height>266</height><depth>1</depth>
</size>
<object><name>table</name>
<bndbox>
<xmin>0</xmin><ymin>0</ymin><xmax>679</xmax><ymax>266</ymax>
</bndbox>
<cells>
<tablecell><x0>11</x0><x1>149</x1><y0>17</y0><y1>100</y1><start_col>0</start_col>
<start_row>0</start_row><end_col>0</end_col><end_row>1</end_row></tablecell>
<tablecell><x0>11</x0><x1>149</x1><y0>100</y0><y1>118</y1><start_col>0</start_col>
<start_row>2</start_row><end_col>0</end_col><end_row>2</end_row></tablecell>
<tablecell><x0>11</x0><x1>149</x1><y0>118</y0><y1>152</y1><start_col>0</start_col>
<start_row>3</start_row><end_col>0</end_col><end_row>4</end_row></tablecell>
<tablecell><x0>11</x0><x1>149</x1><y0>152</y0><y1>169</y1><start_col>0</start_col>
<start_row>5</start_row><end_col>0</end_col><end_row>5</end_row></tablecell>
<tablecell><x0>11</x0><x1>149</x1><y0>169</y0><y1>216</y1><start_col>0</start_col>
<start_row>6</start_row><end_col>0</end_col><end_row>8</end_row></tablecell>
</cells>
</object>
</annotation>
```

		Estimated Future Payouts Under Non-Equity Incentive Plan Awards			All Other Stock Awards: Number of Shares of Stock or Units (#)	All Other Option Awards: Number of Securities Underlying Options (#)	Exercise or Base Price of Option Awards ($/Sh)	Grant Date Fair Value of Stock and Option Awards ($)[3]
Name	Grant Date	Threshold ($)	Target ($)	Maximum ($)				
Aaron Levie[1]	—	—	99,000	—	—	—	—	—
Dan Levin	—	—	165,000	—	—	—	—	—
	5/24/2016	—	—	—	—	300,000	12.09	1,554,240
Dylan Smith[2]	—	—	165,000	—	—	—	—	—
Peter McGoff	—	—	144,000	—	—	—	—	—
	5/24/2016	—	—	—	67,500	—	—	816,075
	5/24/2016	—	—	—	107,500	—	—	1,299,675
Graham Younger ...	5/24/2016	—	337,500	—	60,000	—	—	725,400
	5/24/2016	—	—	—	40,000	—	—	483,600

Figure 2. Shows our approach. Cell detection is done using TOD-Net. Bounding boxes used as an input by the structure recognition model (based on DGCNN [22], which predicts rectilinear adjacencies. These are then collectively used by the post-processing step to generate output XML containing structure).

sub-cells to preserve the complete table structure. These algorithms are robust to input types (scanned images or native digital) and do not generally make assumptions about the layouts. They are data-driven, and easy to fine-tune across different domains. Some methods that use linguistic context were proposed by [2, 18, 5]. Many invoice-specific table extraction models have also been proposed [25, 11].

Recently, many researchers have opted for a graph-based formulation of the problem as a graph is inherently an ideal data structure to model associations between entities [22, 4, 24]. Raja [24] proposed a first end-to-end object detection and graph based model for collective cells detection and structure recognition. Another recent work, GTE-Cell [41], follows a nested approach by first classifying whether a table includes ruling lines or not, and then uses specifically tailored heuristics to identify the table structure. While these methods contribute to significant progress, they make certain assumptions like the availability of accurate word bounding boxes, machine readable PDF documents, and others, as additional inputs [18, 22, 4, 30]. Contrarily, the TabStruct-Net [24] does not make any such assumptions and produces adjacency relations and cell locality information as the output. However, it fails to capture empty cells accurately and, in many cases, results in a significant overlap between detected cells. Further, its structure recognition module failed to correctly identify row/column associations between far-apart cells in case of dense tables.

Given the recent successes in natural object detection [3, 29, 38], and the effectiveness of attention in improving its performance [34, 34, 35, 37], we base our cell detection model on the object detection paradigm. Our work aims to localize low-level table objects better on higher IoU thresholds, including empty cells. Our work also improves long range associations for structure recognition through rectilinear adjacency based formulation.

3. Proposed Method

We formulate the table understanding problem at two levels — *low-level*, i.e., detection of table objects (rows, columns, and cells) and *high-level*, i.e., physical structure recognition by building associations between cells. Most existing methods define table cells as the smallest polygon that encapsulates its content. This has two shortcomings. (i) It misses on the alignment and continuity constraints that are very natural to human cognition; and (ii) it misses on empty cells that usually carry important semantic meanings. Inspired by human cognition, we say that table cells, in addition to completely encapsulating their content, should adhere to alignment [24], continuity and non-overlapping constraints, which in-turn makes it easier to locate table columns and rows as independent objects.

As discussed in Section 1, many existing methods assume pre-located cell content and target only high-level structure understanding. Usually, table cells' coordinates are obtained by parsing corresponding PDF/LaTeX documents which may not always be available. Several methods also use OCR tools to extract cell contents, resulting in the loss of intra-cell associations and structural alignment. Further, the absence of cell content makes it difficult to consider empty cells for structure recognition. In many real-world documents, empty cells carry a semantic meaning and must be associated with the table to obtain an accurate table structure. Not taking them into account might lead to false negatives and, in-turn, incorrect structure [24]. To localize table cells, we experiment by solving cell detection directly and through the intersection of predicted rows and columns. After locating all cells, we build rectilinear associations between every pair by formulating the problem as a graph.

Our solution progresses in three steps, as shown in Figure 2, — (i) table cell detection using visual cues, (ii) structure recognition by forming rectilinear associations through a graph-based formulation, and (iii) collating bounding boxes and rectilinear associations to obtain row and column spanning values for every table cell.

3.1. Cell Detection

We aim to detect table cells in two ways — (i) by locating them as independent objects and (ii) by first locating rows and columns as independent objects and then using intersections to obtain cell coordinates. We target row, column, and cell detection as object detection tasks using our Table Object Detection Network (TOD-Net shown in Figure 2), built on top of the cell detection network of TabStruct-Net [24, 10]. Our augmentations to the existing architecture aim to model the constraints associated with table objects to ensure adjacent cells' continuity and non-overlap. We use sparse channel weights on the ROI aligned feature maps to predict table objects' bounding boxes (cells, rows, and columns). We also formulate the problem as min-max optimization using adaptable loss weights for the three structural regularizers — alignment loss [24], continuity loss, and overlap loss.

Notations: Let $\mathbb{X}$ denote the set of table images; $SR(i)$, $SC(i)$, $ER(i)$, and $EC(i)$ represent start-row, start-column, end-row, and end-column indices respectively; and x_{1i}, y_{1i}, x_{2i} and y_{2i} represent bounding box coordinates start-x, start-y, end-x, and end-y, respectively of the object i. i and j denote two table objects (row/column/cell). L_m denotes the sum of RPN class loss, RPN bounding box regressor loss, Mask R-CNN class loss, Mask R-CNN bounding box regressor loss, and mask loss. L_{al}, L_{cl}, L_{ol}^x, and L_{ol}^y represent alignment loss, continuity loss, and overlap losses along X and Y directions respectively; and W_{al}, W_{cl}, W_{ol}^x, and W_{ol}^y represent corresponding learnable weights.

Continuity Loss: The intuition behind adding continuity loss is that horizontally adjacent objects should end and start at the same x-coordinate and vertically adjacent objects end and start at the same y-coordinate. Continuity loss is given in Eq. (1)

$$\begin{aligned} L_{cl}^{row} &= \sum_{i,j} ||y_{1i} - y_{2j}||_2^2 \cdot \ \mathbb{I}\big(SR(i) == ER(j) + 1\big) \\ L_{cl}^{col} &= \sum_{i,j} ||x_{1i} - x_{2j}||_2^2 \ \cdot \mathbb{I}\big(SC(i) == EC(j) + 1\big) \\ L_{cl} &= L_{cl}^{row} + L_{cl}^{col}. \end{aligned} \tag{1}$$

This loss helps to predict well-aligned coordinates by accurately capturing the background or non-text region associated with objects that are significantly wider or longer than the text region contained in them.

Overlapping Loss: We introduce overlapping loss as an L2 regularizer to minimize overlapping regions between every pair of predicted table objects. During the calculation, the overlap of an object with itself does not account for the loss. Further, it is computed independently along X and Y directions (as given in Eq. 2).

$$\begin{aligned} L_{ol}^x &= \sum_{i,j} ||(min(x_{2i}, x_{2j}) - max(x_{1i}, x_{1j})||_2^2 \cdot \mathbb{I}\big(i! = j\big), \\ L_{ol}^y &= \sum_{i,j} ||(min(y_{2i}, y_{2j}) - max(y_{1i}, y_{1j}))||_2^2 \cdot \mathbb{I}\big(i! = j\big) \end{aligned} \tag{2}$$

Trainable Loss Weights: We incorporate trainable loss weights for four different structure components as regularizers (alignment, continuity, and overlap loss along X and Y directions) for every region of interest (ROI) independently such that the weights add up to one. This allows for a dynamic emphasis on different structural constraints for different ROIs based on their visual characteristics during training. We model the optimization problem as a min-max optimization problem as follows:

$$\begin{aligned} \mathbb{L}(\mathbb{X}, \theta_m, \theta_W) = min_{\theta_m}\big(\ L_m(\theta_m)\big) \ + \max_{\theta_W}\big(\\ W_{al}(\theta_W) \cdot L_{al}(\theta_m) \ + \ W_{cl}(\theta_W) \cdot L_{cl}(\theta_m) + \\ W_{ol}^x(\theta_W) \cdot L_{ol}^x(\theta_m) \ + \ W_{ol}^x(\theta_W) \cdot L_{ol}^y(\theta_m) \ \big) \\ \ni W_{al} \ + \ W_{cl} \ + \ W_{ol}^x \ + \ W_{ol}^y \ = \ 1 \end{aligned} \tag{3}$$

Since we need to minimize the objective loss (as given in Eq. (3)) over θ_m and maximize over θ_W, the parameter updates are given by the following Eq. (4)

$$\begin{aligned} \theta_m^{t+1} &= \theta_m^t \ - \ \eta \ \cdot \ \nabla_{\theta_m^t}\big(\mathbb{L}(\mathbb{X}, \theta_m^t, \theta_W^t)\big) \\ \theta_W^{t+1} &= \theta_W^t \ + \ \eta \ \cdot \ \nabla_{\theta_W^t}\big(\mathbb{L}(\mathbb{X}, \theta_m^t, \theta_W^t)\big), \end{aligned} \tag{4}$$

where η is the learning rate. Formulation based on a min-max optimization problem using trainable loss weights (by allowing for weighting different regularizers differently based on RoI's visual features) not only improves optimization speed, but also proves useful during post-processing. We use the predicted values of loss weights during the test time to identify and correct overlapping or misaligned cells. Our experiments suggest that high overlapping loss weights were observed during test time for dense table images. Similarly, high alignment values and continuity losses were observed for multi-column or multi-row spanning header cells where the text was not aligned in the center.

Channel Attention: To detect table objects' start and end coordinates, specific visual patterns such as separator lines or non-text regions need to be present. These visual patterns differ significantly from general object detection problems where different shaped edges and textures are essential to distinguish different types of objects. For the detection of table objects, the distinguishing visual clues occur in particular regions of every ROI. In order to localize table cells, specific set of visual features contribute. For example, a column (or a row) would start or end at an x (or y)-coordinate where around that region, either a vertical (or a horizontal) separator or non-text/background is observed along the length (or width) of the image. This motivates us to incorporate L1-regularized channel-wise attention to look for specific sparse patterns to detect cell bounding boxes accurately. The attention-mechanism we use is based on the architecture proposed by [19] and is shown in Figure 2.

3.2. Structure Recognition

We formulate the table structure recognition as a graph learning problem similar to [22]. However, instead of creating row and column adjacency matrices, we create four rectilinear matrices such as left (M_l), right (M_r), top (M_t), and bottom (M_b) $\in R^{n \times n}$, where n denotes the number of detected cells. For the top rectilinear matrix, M_t, the element at $M_t(i, j)$ indicates whether cell j is at the top of cell i. Similarly, we create left M_l, right M_r, and bottom M_b matrices. Formulating the problem in this way allows for better capturing of long-range dependencies for dense tables particularly. We use four instances of the DGCNN architecture proposed in [22] to predict the four rectilinear matrices. The DGCNN consists of three components — (i) a visual network to generate a visual feature map corresponding to the input table image, (ii) an interaction network to capture associations between cells from the visual features and coordinates of table cells, and (iii) a classification network to determine if a pair of table cells are left/right/top/bottom adjacent. The training happens in two steps. In the first step, we use ground-truth boxes, and in the second step, we fine-tune the models using predictions of TOD-Net on the training dataset. The training adjacencies are obtained by identifying the largest overlapping ground truth cell corresponding to the prediction.

3.3. Post-processing

Firstly, we fine-tune the predicted cell bounding boxes using Tesseract's [28] word bounding boxes to ensure that the predicted cell boundary region does not pass through any text region. Once cell bounding boxes and rectilinear adjacency matrices are obtained, the next step is to figure out row and column spanning values for every cell. The maximum count of left and right adjacencies is obtained recursively to obtain row span for cell i. Similarly, to obtain column span for cell i, the maximum count of top and bottom adjacencies is obtained recursively. Finally, start-row (SR), end-row (ER), start-column (SC), and end-column (EC) indices for every cell are obtained by sorting the coordinates based on start-x and end-x coordinates along with the row and column spans obtained using the rectilinear adjacency matrices. The use of rectilinear adjacencies accounted for reduced use of heuristics and improved F1 scores for structure recognition. Our final output comes out as an XML that contains bounding boxes along with the row and column spans for every cell given a table image.

Dataset	Document Domain	Alignment Constraint	#Train Image	#Test Image
ICDAR-2013	Business	×	-	156
UNLV	Business	✓	-	558
cTDaR	Business	×	600	150
SciTSR	Scientific	×	12K	3K
Table2Latex	Scientific	×	447K	9K
TableBank	Scientific	×	145K	1K
PubTabNet	Scientific	×	420K	40K
FinTabNet	Business	×	91K	10K
TUCD (our)	Business	✓	-	4.5K

Table 1. Presents statistics of datasets for table structure recognition. Only TableBank [17] is dedicated for logical table structure recognition. All other datasets are used for physical table structure recognition.

3.4. Datasets

Most datasets [22, 4, 6, 17, 42, 41] use words or cell content as low-level entities to build inter-tabular relationships. Similarly, there exist inconsistencies in the datasets for predicting the physical or logical structure of tables. This presents a fundamental challenge to evaluate and compare various methods for table structure recognition directly. [22, 4, 17, 42] introduced many large-scale automatically generated datasets, but they do not accurately represent real-world complex tables as seen in the business documents [41, 27, 8]. Another matter of concern is the style of annotation. As humans, we think of tables adhering to specific structural and alignment constraints — (i) cells belonging to the same row should start and end at the same start-y and end-y coordinates respectively, (ii) cells belonging to the same column should start and end at the same start-x and end-x coordinates respectively, (iii) cells starting at column i should have the same start-x coordinate as the end-x coordinate of column $i-1$, (iv) cells starting at row i should have the same start-y coordinate as the end-y coordinate of row $i-1$, (v) no overlap between any pair of table cells. Presently, UNLV [27] is the only dataset where ground-truth preserves this inherent structural alignment between cells. However, this dataset is limited in size, language, and domain variations for evaluating a deep learning-based method. Other datasets [4, 42, 41, 8] have annotations such that a cell's bounding box is the smallest rectangle that encapsulates its content. This leads to non-annotation for empty cells and loss of alignment between cells in the same and adjacent rows/columns.

TUCD dataset is dedicated to evaluation of cells detection and structure recognition for business documents. It consists of 4500 table images collected from the publicly available annual reports in English and non-English languages (e.g., French, Japanese, Russian, and others) of more than ten years from twenty-nine different companies[1]. The ground truth XML for a table image contains the coordinates of bounding boxes of cells and their row and column spans. Table 1 lists the statistics of different structure recognition datasets available for training and testing.

3.5. Training and Evaluation

We use FinTabNet [41] dataset to train TOD-Net for cell, row, and column detection. Since FinTabNet has bounding boxes wrapped around the cell's content, we pre-process the ground truth to obtain cell level coordinates (refer supplementary paper)[2]. The resulting dataset follows all the constraints that we model in the TOD-Net. For evaluation also, we pre-process ICDAR-2013 [8], cTDaR [7], SciTSR [4], PubTabNet [42] and FinTabNet [41] datasets before computing IoU with the corresponding predictions. Since UNLV [27] and TUCD datasets already have annotations for cells adhering to alignment constraints, we directly used them for evaluation. Further, during training and evaluation, we use the non-maximal suppression threshold of 0.8 during proposal generation to reduce the false negatives substantially. We train TSR-Net in two steps: In the first stage, we use pre-processed ground-truth cell boxes and corresponding start-row, start-column, end-row, and end-column indices to generate target rectilinear adjacency matrices. In the second stage, we generate predictions of the training set using TOD-Net to compute its overlap with the ground-truth to find start-row, start-column, end-row, and end-column indices for every predicted box. We accordingly generate target rectilinear adjacency matrices for training on the predicted boxes.

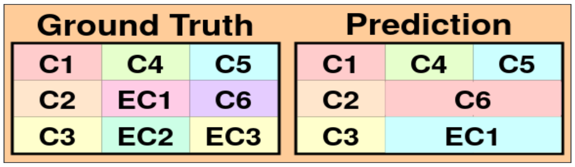


Figure 3. Shows sample ground truth and predicted bounding boxes of cells for evaluation. Assume Cs to be cells with content and ECs to be cells without content. Also, assume detection of table cells merges EC1 and C6 in row 2 and EC2 and EC3 in row 3. Our proposed evaluation criteria additionally penalize (EC1, EC2) and (EC2, EC3) as false negatives.

3.6. Evaluation Protocol

In literature, researchers [8, 27, 4] use precision, recall, and F1 scores to evaluate the performance of table's physical structure recognition. Adjacency relations for every true positive cell are generated with their horizontal and vertical neighbors to assess structure recognition performance. The predicted relation list is then compared with the ground truth list to calculate precision, recall, and F1 scores. However, these criteria do not consider empty cells that are not surrounded by non-empty cells to calculate performance scores. Since most existing methods use pre-located table cells as inputs, this does not cause any problem. However, as a result of cell detection, these empty cells might get merged with neighboring cells containing content or false positives, disturbing the overall table structure (as shown in Figure 3). Henceforth, for the end-to-end structure recognition of given table images only, we suggest taking into account empty cells to calculate precision, recall, and F1 scores correctly. For table object (row, column, and cell) detection (both empty and with content), we calculate precision, recall, and F1 scores for an IoU threshold of 0.6.

4. Results

This work presents a comprehensive analysis of results to understand the impact of architectural designs, modifications to the evaluation criteria, and optimization characteristics. For this purpose, we provide a four-fold analysis - comparative analysis with existing methods in the literature, analysis on varying IoU thresholds for cell detection, an ablation study showing the effectiveness of design choices and impact of loss weights on optimization speed.

Comparative Analysis Table 2 shows results comparing our method against previously published on ICDAR-2013, SciTSR, ICDAR-19 and TUCD datasets. Please note

[1] TUCD dataset is available at https://github.com/sachinraja13/TUCD

[2] Please refer supplementary material for dataset preprocessing, post-processing, implementation and additional quantitative and qualitative results.

			Average Over Test Set														
	Training		**ICDAR-2013**			**SciTSR**			**SciTSR Comp**			**ICDAR-19**			**TUCD**		
Method	**Dataset**	**EC**	**P↑**	**R↑**	**F1↑**	**P↑**	**R↑**	**F1↑**	**P↑**	**R↑**	**F1↑**	**P↑**	**R↑**	**F1↑**	**P↑**	**R↑**	**F1↑**
DeepDeSRT [26]	ICDAR-13	SEC	*0.96*	*0.87*	*0.91*	-	-	-	-	-	-	-	-	-	-	-	-
SPLERGE(H) [30]	Private		*0.96*	*0.95*	*0.95*	-	-	-	-	-	-	-	-	-	-	-	-
SPLIT [30]	Private		*0.87*	*0.87*	*0.87*	0.92	0.97	0.97	0.91	0.88	0.90	0.70	0.67	0.69	0.87	0.86	0.86
TabStruct-Net [24]	SciTSR		*0.92*	*0.90*	*0.91*	*0.93*	*0.91*	*0.92*	*0.91*	*0.88*	*0.90*	*0.60*	*0.57*	*0.58*	0.90	0.89	0.90
GTE-Cell [41]	FinTabNet		*0.96*	*0.97*	*0.96*	-	-	-	-	-	-	-	-	-	-	-	-
SEM [40]	SciTSR		-	-	-	*0.98*	*0.97*	*0.97*	*0.97*	*0.95*	*0.96*	-	-	-	-	-	-
LGPMA [23]	SciTSR		*0.93*	*0.98*	*0.95*	***0.98***	***0.99***	***0.99***	***0.97***	***0.99***	***0.98***	-	-	-	-	-	-
DeepDeSRT [26]	FinTabNet	SEC	0.82	0.80	0.81	0.87	0.85	0.86	0.85	0.83	0.84	0.55	0.51	0.53	0.73	0.70	0.72
DGCNN† [22, 24]	FinTabNet		0.94	0.93	0.94	0.91	0.89	0.90	0.89	0.88	0.89	0.73	0.70	0.71	0.89	0.87	0.88
DGCNN‡ [22, 24]	FinTabNet		0.96	0.95	0.96	0.91	0.90	0.91	0.90	0.89	0.89	0.76	0.73	0.74	0.92	0.91	0.91
TabStruct-Net [24]	FinTabNet		0.95	0.94	0.95	0.90	0.89	0.90	0.88	0.87	0.87	0.76	0.73	0.75	0.91	0.90	0.90
Ours†	FinTabNet		0.95	0.95	0.95	0.92	0.91	0.92	0.92	0.90	0.91	0.72	0.70	0.71	0.91	0.90	0.91
Ours‡	FinTabNet		**0.98**	**0.97**	**0.97**	0.94	0.92	0.93	0.93	0.89	0.91	**0.77**	**0.76**	**0.77**	**0.94**	**0.93**	**0.93**
DeepDeSRT [26]	FinTabNet	NEC	0.74	0.71	0.73	0.82	0.80	0.81	0.80	0.79	0.79	0.53	0.48	0.50	0.70	0.68	0.69
SPLIT [30]	Private		0.83	0.81	0.82	0.89	0.87	0.88	0.87	0.87	0.87	0.68	0.66	0.67	0.82	0.81	0.81
DGCNN† [22, 24]	FinTabNet		0.87	0.85	0.86	0.89	0.87	0.88	0.87	0.85	0.86	0.69	0.67	0.68	0.86	0.85	0.85
DGCNN‡ [22, 24]	FinTabNet		0.90	0.89	0.89	0.88	0.85	0.86	0.86	0.84	0.85	0.71	0.69	0.70	0.89	0.88	0.89
TabStruct-Net [24]	SciTSR		0.89	0.87	0.88	0.90	0.87	0.88	0.88	0.86	0.87	0.54	0.49	0.51	0.84	0.83	0.83
TabStruct-Net [24]	FinTabNet		0.90	0.87	0.89	0.88	0.85	0.86	0.86	0.84	0.85	0.70	0.69	0.70	0.88	0.86	0.87
Ours†	FinTabNet		0.91	0.90	0.90	0.90	0.86	0.88	0.88	0.84	0.86	0.70	0.67	0.68	0.90	0.88	0.89
Ours‡	FinTabNet		**0.93**	**0.92**	**0.92**	**0.91**	**0.88**	**0.89**	**0.89**	**0.87**	**0.88**	**0.73**	**0.72**	**0.72**	**0.92**	**0.91**	**0.92**

Table 2. Compares various methods for table structure recognition on ICDAR-2013, SCI-TSR, SCI-TSR COMP, ICDAR-19 and TUCD datasets. Scores in italics are directly reported from corresponding papers. For others, we use open source implementations and pre-trained models released by authors. For DeepDeSRT [26], we use our implementation. EC: indicates evaluation criteria, SEC: indicates standard evaluation criteria, and NEC: indicates new evaluation criteria. P: indicates precision, R: indicates recall, and F1: indicates F1 score. TOD-Net†: indicates TOD-Net for direct cell detection and TOD-Net‡: indicates cell detection using intersection of TOD-Net results row and column predictions, DGCNN† indicates TOD-Net†+DGCNN+PP, DGCNN‡ indicates TOD-Net‡+DGCNN+PP TS-Net indicates TabStruct-Net, Ours† indicates TOD-Net†+TSR+PP, Ours‡ indicates TOD-Net‡+TSR+PP and (H) indicates dataset specific heuristics. For comparison on ICDAR-2013 using SEC, ICDAR-2013 text-based evaluation was used. All other results are based on a fixed IoU threshold of 0.6. For the NEC, we additionally consider empty cells for evaluation.

that in the first section of the table with evaluation using Standard Evaluation Criteria (SEC), we use ICDAR-2013 text-based measure for ICDAR-2013 dataset. On the contrary, corresponding SEC, we use IoU overlap based ICDAR-2019 evaluation criterion on SciTSR, ICDAR-19 and TUCD datasets. Further, for the second section of the table, that uses New Evaluation Criteria (NEC), we modify the IoU based ICDAR-2019 evaluation to additionally take into account adjacency relations between empty-empty and empty-non empty cells. For evaluating ICDAR-2013 dataset using NEC, we modify the ground truth to obtain cell-level boxes (as explained in Section 3.5) and extend those to full rows and columns to obtain bounding box coordinates for empty cells (assuming no empty cells are multi-row/column spanning). Details of this step are provided in the supplementary section. For a fair comparison of our method against DGCNN [22], we use TOD-Net to obtain cell bounding boxes, obtain row and column adjacency matrices using DGCNN [22] and use the open-source post-processing provided by [24]. In order to compare our method against others on TUCD dataset, we develop our implementation of DeepDeSRT [26], and use open source implementations of DGCNN (TIES) [22], SPLERGE [30], and TabStruct-Net [24]. For others, we directly report results from the corresponding papers. From the table, it is evident that formulating the problem using rectilinear adjacencies instead of row/column adjacency avoids errors in long visual ranges, relaxes heuristics in the post-processing method. Our method outperforms previous state-of-the-art on all three datasets by a reasonable difference of average F1-score on structure recognition. We further observe that empty cells account for an average of 12.3% across UNLV and ICDAR-2013 datasets, where our method outperforms TabStruct-Net by 4.2% F1 score.

Our solution however fails for very sparse tables where most of the cells are empty. We will add some qualitative

Method	IoU	FinTabNet	ICDAR-13	Sci-TSR	TUCD
		TSR-F1↑	TSR-F1↑	TSR-F1↑	TSR-F1↑
TS-Net	0.5	0.898	0.904	0.876	0.900
Ours†		0.906	0.903	0.880	0.889
Ours‡		**0.944**	**0.904**	**0.894**	**0.918**
TS-Net	0.6	0.848	0.886	0.864	0.871
Ours†		0.892	0.903	0.878	0.889
Ours‡		**0.920**	**0.904**	**0.894**	**0.918**
TS-Net	0.7	0.704	0.720	0.682	0.722
Ours†		0.802	0.820	0.746	0.797
Ours‡		**0.868**	**0.852**	**0.823**	**0.839**
TS-Net	0.8	0.496	0.597	0.565	0.582
Ours†		0.561	0.675	0.637	0.659
Ours‡		**0.680**	**0.748**	**0.714**	**0.735**
TS-Net	0.9	0.120	0.292	0.255	0.289
Ours†		0.325	0.307	0.296	0.301
Ours‡		**0.404**	**0.454**	**0.368**	**0.408**

Table 3. Shows the comparison between the performances of the proposed network and TabStruct-Net (TS-Net) [24] on cell detection and table structure recognition of dataset over various IoU thresholds.TSR: indicates table structure recognition. We use FinTabNet [41] dataset for training.

examples in the supplementary material. Since rectilinear adjacencies are predicted between every pair of cells, inference time is in the order of square of number of cells located. For table images with 20 cells, inference time is about 10 seconds which goes upto 50 seconds for images with 200 cells.

F1 based on Varying IoU Thresholds For table cell detection, the IoU threshold becomes imperative as the penalty for loss of content or additional content detected from a localized table cell is high. Higher IoU also accounts for better structure recognition performance. Hence, a method's robustness can be established based on its performance under a higher IoU threshold. For this purpose, we evaluate the previously established benchmark [24] with our approach on IoU thresholds varying from 0.5 up to 0.9 as shown in Table 3 according to our updated evaluation criteria that take into account empty cells present along the table extreme boundary regions.

Ablation Study Table 4 shows the ablation study of various enhancements to our TOD-Net. We observe that the addition of continuity loss improved the average F1 score by 0.8%. It especially proved helpful for table cells having a varying amount of text in table headers. For text consisting of large empty spaces with a very little text region, continuity loss helped detect the boxes that adhere to the inherent table alignment. We further observed that the addition of pairwise overlapping loss improved precision by 1.1% and channel-wise multiplication of sparse channel weights further improved detection performance by 2.1%. Also, we observe that with the same weight initialization, the model with dynamic loss weights converges 15% faster and slightly better by 0.4%.

Method	Cell Detection		
	P↑	R↑	F1↑
Mask R-CNN+AL	0.880	0.862	0.871
Mask R-CNN+AL+CL	0.891	0.868	0.879
Mask R-CNN+AL+CL+OL	0.907	0.873	0.890
Mask R-CNN +AL+ CL+OL+ROI_Att.	0.922	0.900	0.911
Mask R-CNN+AL+ CL+OL+ROI_Att.+LossWT	**0.926**	**0.904**	**0.915**

Table 4. Shows the ablation study for cell detection on various structural constraints on baseline (Mask R-CNN+AL) [24]. We use new evaluation criteria with IoU threshold = 0.6. TOD: indicates table object detection, AL: indicates alignment loss, CL: indicates continuity loss, OL: indicates overlapping loss, ROI_Att.: indicates ROI attention, and LossWT: indicates loss weights. We use FinTabNet [41] dataset for training and evaluation.

5. Conclusion

Our approach advances both the formulation and the empirical performances compared to the state of the art methods. Major contributions include: (i) a formulation possibly closer to how human perceives tables (ii) architectural improvements to model problem-specific constraints, (iii) an adaptation of optimization, (iv) a novel TUCD dataset for evaluation and (iv) empirical evaluation extending the analysis to high IoU thresholds that improve practical usability.

Our work will advance the table understanding literature with immediate effect for better information extraction from business documents. We also believe, our insights in analyzing images with dense structured objects will impact wider categories of images captured in industrial vision setting, and crowded outdoor. Also, our dataset and improved evaluation can serve for a more robust evaluation of table structure. Further, the reasoning behind using trainable loss weights could be extended to niche domain specific problems (understanding of graphs/charts and establishing correct reading order from document images).

Acknowledgment

This work is partly supported by MEITY, Government of India.

References

[1] Darshan Adiga, Shabir Ahmad Bhat, Muzaffar Bashir Shah, and Viveka Vyeth. Table structure recognition based on cell relationship, a bottom-up approach. In *RANLP*, 2019.

[2] Junwei Bao, Duyu Tang, Nan Duan, Zhao Yan, Yuanhua Lv, Ming Zhou, and Tiejun Zhao. Table-to-text: Describing table region with natural language. In *AAAI*, 2018.

[3] Zhaowei Cai and Nuno Vasconcelos. Cascade R-CNN: Delving into high quality object detection. In *CVPR*, 2018.

[4] Zewen Chi, Heyan Huang, Heng-Da Xu, Houjin Yu, Wanxuan Yin, and Xian-Ling Mao. Complicated table structure recognition. *arXiv*, 2019.

[5] Li Deng, Shuo Zhang, and Krisztian Balog. Table2Vec: Neural word and entity embeddings for table population and retrieval. In *SIGIR*, 2019.

[6] Yuntian Deng, David Rosenberg, and Gideon Mann. Challenges in end-to-end neural scientific table recognition. In *ICDAR*, 2019.

[7] L. Gao, Y. Huang, H. Déjean, J. Meunier, Q. Yan, Y. Fang, F. Kleber, and E. Lang. ICDAR 2019 competition on table detection and recognition (cTDaR). In *ICDAR*, 2019.

[8] Max Göbel, Tamir Hassan, Ermelinda Oro, and Giorgio Orsi. ICDAR 2013 table competition. In *ICDAR*, 2013.

[9] E Green and M Krishnamoorthy. Recognition of tables using table grammars. In *Annual Symposium on Document Analysis and Information Retrieval*, 1995.

[10] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *CVPR*, 2017.

[11] Martin Holeček, Antonín Hoskovec, Petr Baudiš, and Pavel Klinger. Line-items and table understanding in structured documents. *arXiv*, 2019.

[12] Jianying Hu, Ramanujan S Kashi, Daniel P Lopresti, and Gordon Wilfong. Medium-independent table detection. In *Document Recognition and Retrieval VII*, 1999.

[13] Katsuhiko Itonori. Table structure recognition based on textblock arrangement and ruled line position. In *ICDAR*, 1993.

[14] Saqib Ali Khan, Syed Muhammad Daniyal Khalid, Muhammad Ali Shahzad, and Faisal Shafait. Table structure extraction with Bi-directional Gated Recurrent Unit networks. In *ICDAR*, 2019.

[15] Thomas G Kieninger. Table structure recognition based on robust block segmentation. In *Document Recognition V*, 1998.

[16] Nataliya Le Vine, Matthew Zeigenfuse, and Mark Rowan. Extracting tables from documents using conditional generative adversarial networks and genetic algorithms. In *IJCNN*, 2019.

[17] Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, Ming Zhou, and Zhoujun Li. TableBank: Table benchmark for image-based table detection and recognition. In *ICDAR*, 2019.

[18] Kyosuke Nishida, Kugatsu Sadamitsu, Ryuichiro Higashinaka, and Yoshihiro Matsuo. Understanding the semantic structures of tables with a hybrid deep neural network architecture. In *AAAI*, 2017.

[19] Mehrdad Noori, Ali Bahri, and Karim Mohammadi. Attention-guided version of 2d UNet for automatic Brain Tumor segmentation. In *ICCKE*, 2019.

[20] 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. In *ICDAR*, 2019.

[21] 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. In *CVPRW*, 2020.

[22] Shah Rukh Qasim, Hassan Mahmood, and Faisal Shafait. Rethinking table parsing using graph neural networks. In *ICDAR*, 2019.

[23] Liang Qiao, Zaisheng Li, Zhanzhan Cheng, Peng Zhang, Shiliang Pu, Yi Niu, Wenqi Ren, Wenming Tan, and Fei Wu. Lgpma: Complicated table structure recognition with local and global pyramid mask alignment. *arXiv preprint arXiv:2105.06224*, 2021.

[24] Sachin Raja, Ajoy Mondal, and C. V. Jawahar. Table structure recognition using top-down and bottom-up cues. In *ECCV*, 2020.

[25] Pau Riba, Anjan Dutta, Lutz Goldmann, Alicia Forńes, Oriol Ramos, Llaďos, and Josep. Table detection in invoice documents by graph neural networks. In *ICDAR*, 2019.

[26] 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 *ICDAR*, 2017.

[27] Asif Shahab, Faisal Shafait, Thomas Kieninger, and Andreas Dengel. An open approach towards the benchmarking of table structure recognition systems. In *DAS*, 2010.

[28] Ray Smith. An overview of the Tesseract OCR engine. In *ICDAR*, 2007.

[29] Mingxing Tan, Ruoming Pang, and Quoc V Le. Efficientdet: Scalable and efficient object detection. In *CVPR*, 2020.

[30] Christopher Tensmeyer, Vlad Morariu, Brian Price, Scott Cohen, and Tony Martinezp. Deep splitting and merging for table structure decomposition. In *ICDAR*, 2019.

[31] Scott Tupaj, Zhongwen Shi, C Hwa Chang, and Hassan Alam. Extracting tabular information from text files. *EECS Department, Tufts University, Medford, USA*, 1996.

[32] Nam Van Nguyen, Hanh Vu, Arthur Zucker, Younes Belkada, Hai Van Do, Doanh Ngoc-Nguyen, Thanh Tuan Nguyen Le, and Dong Van Hoang. Table structure recognition in scanned images using a clustering method. In *ICINIS*, 2020.

[33] Nancy Xin Ru Wang, Douglas Burdick, and Yunyao Li. TableLab: An interactive table extraction system with adaptive deep learning. *arXiv*, 2021.

[34] Qilong Wang, Banggu Wu, Pengfei Zhu, Peihua Li, Wangmeng Zuo, and Qinghua Hu. ECA-Net: Efficient channel attention for deep convolutional neural networks. In *CVPR*, 2020.

[35] Xudong Wang, Zhaowei Cai, Dashan Gao, and Nuno Vasconcelos. Towards universal object detection by domain attention. In *CVPR*, 2019.

[36] Yalin Wang, Ihsin T Phillips, and Robert M Haralick. Table structure understanding and its performance evaluation. *Pattern Recognition*, 2004.

[37] Zhonghua Wu, Qingyi Tao, Guosheng Lin, and Jianfei Cai. Exploring bottom-up and top-down cues with attentive learning for webly supervised object detection. In *CVPR*, 2020.

[38] Hang Xu, Lewei Yao, Wei Zhang, Xiaodan Liang, and Zhenguo Li. Auto-FPN: Automatic network architecture adaptation for object detection beyond classification. In *ICCV*, 2019.

[39] Wenyuan Xue, Qingyong Li, and Dacheng Tao. ReS2TIM: Reconstruct syntactic structures from table images. In *ICDAR*, 2019.

[40] Zhenrong Zhang, Jianshu Zhang, and Jun Du. Split, embed and merge: An accurate table structure recognizer. *arXiv preprint arXiv:2107.05214*, 2021.

[41] 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 *WACV*, 2021.

[42] Xu Zhong, Elaheh ShafieiBavani, and Antonio Jimeno Yepes. Image-based table recognition: data, model, and evaluation. In *ECCV*, 2020.