Deep Splitting and Merging for Table Structure Decomposition

Chris Tensmeyer, Vlad I. Morariu, Brian Price, Scott Cohen
Adobe Research
San Jose, USA
{*tensmeye,morariu,bprice,schoen*}*@adobe.com*

Tony Martinez
Dept. of Computer Science
Brigham Young University
Provo, USA
martinez@cs.byu.edu

***Abstract*—Given the large variety and complexity of tables, table structure extraction is a challenging task in automated document analysis systems. We present a pair of novel deep learning models (Split and Merge models) that given an input image, 1) predicts the basic table grid pattern and 2) predicts which grid elements should be merged to recover cells that span multiple rows or columns. We propose projection pooling as a novel component of the Split model and grid pooling as a novel part of the Merge model. While most Fully Convolutional Networks rely on local evidence, these unique pooling regions allow our models to take advantage of the global table structure. We achieve state-of-the-art performance on the public ICDAR 2013 Table Competition dataset of PDF documents. On a much larger private dataset which we used to train the models, we significantly outperform both a state-of-the-art deep model and a major commercial software system.**

***Keywords*-Table Structure Extraction; Deep Learning; Convolutional Neural Network**

I. Introduction

Tables are commonly used to present structured data in documents. Automatically determining the structure of tables, i.e., the locations of cells and how they are organized into columns and rows, is a crucial part of document understanding and information extraction.

This work focuses on documents as images and in Portable Document Format (PDF) that contain tables with no structural annotations. Images, such as document scans, are composed of pixels that visually reproduce the document, but do not have any structural information. While PDFs are composed of discrete drawing commands for text and line art, table structure is not explicitly encoded in these commands. For example, a single text drawing command may be used to render part of a cell, an entire cell, parts of multiple cells, or multiple entire cells. Such ambiguities make it non-trivial to extract table structure from PDF drawing commands.

Several factors complicate table structure parsing (see [1] for an overview). Lined tables are generally easier than unlined tables [2] because they have explicit visual cues that delimit cells, columns, and rows. Other tables are partially lined or use shading to denote boundaries. Tables may have a complex and hierarchical header structure or may have no headers. Processing tables with multi-line cells involves determining if vertically adjacent text lines are part of the same cell or not. Additionally, cells that span multiple columns and rows add to table complexity.

Many previous works that extract table structure are rule based systems that analyze PDF drawing commands [2]–[5], i.e., they don't directly analyze the document image. While heuristic methods have previously outperformed learning methods on PDFs, they cannot be directly applied to images. Furthermore, heuristic rules have many parameters that must be tuned as tables vary widely in structure, visual appearance, and complexity.

Recently, deep learning has been proposed to learn table structure directly from images [6]. The Deep DeSRT model [6] uses a neural network originally designed for semantic segmentation of natural scenes and thus primarily uses local information to classify pixels. However, local information is often insufficient to identify table cell boundaries because both cells and cell separators contain large regions of white space. Furthermore, cell boundaries in many unlined tables are visually indicated by globally consitent text justification (e.g., all cells in a column are left justified to the column boundary). To solve these issues, our proposed deep models perform pooling over large image regions to allow discriminative information to propagate across the whole image.

In this work, we propose a pair of deep learning models that predicts table structure from well-cropped table images. The *Split* model predicts the fine grid structure of the table, and the *Merge* model predicts where cells span multiple columns or rows. Together, we refer to the combined Split and Merge models as SPLERGE. The Split model uses novel pooling regions, termed *projection pooling*, where features are pooled along rows and columns of pixels to help learned features propagate across large images. Thus, pixels in the same row (column), but on opposite sides of the image, can exchange their local information on where the row (column) boundaries are. The output of this model is the grid structure of the table, ignoring spanning cells. The Merge model takes as input the original table image and predicts which elements of the previously predicted grid need to be merged to recover spanning cells. The novel contribution of the Merge model is *grid pooling*, where pooling regions are determined by the grid output of the Split model. This allows for local pixels features to easily propagate across large cells and

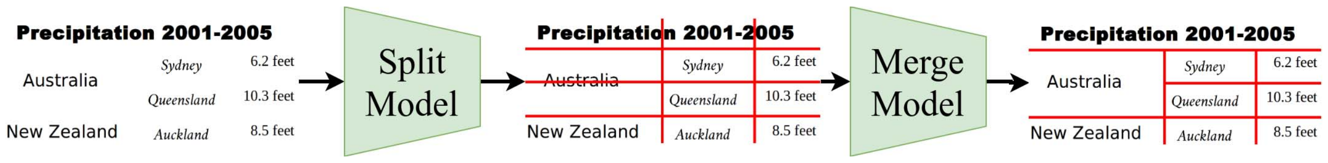


Figure 1. Overview of SPLERGE. First the Split model predicts the basic grid of the table, ignoring cells that span multiple rows or columns. Then the Merge model predicts which grid elements should be merged to recover spanning cells.

allow features to be exchanged among neighboring cells.

We achieve state-of-the-art performance on the ICDAR 2013 Table Competition dataset [2], obtaining an adjacency relationship F-measure of 95.15% compared to the best published result of 94.6% [2]. We also evaluate SPLERGE on a much larger and diverse private dataset of table images, significantly outperforming a state-of-the-art deep learning model and a major commercial software system.

II. Related Works

While much work has focused on table localization, fewer works have attempted to extract table structure.

The ICDAR 2013 Table Competition [2] compared many systems for table structure recognition, including the current state-of-the-art method by Nurminen [3]. This method performs edge detection on the rendered PDF image (filtering out edges due to text) to find junctions and rectangular areas. Then it follows a series of heuristic steps to assign text to rows, identifying likely columns and headers, revising rows, merging columns, find spanning cells, etc. Shigarov et al. [5] later proposed a simpler set of heuristics based on bottom-up merging of PDF components. The TEXUS system [4] performs simple table structure parsing by considering each text chunk to be a cell. Compared to these methods, we do not require heuristics and can operate on both PDF and image input tables.

The DeepDeSRT [6] learning model formulates table structure extraction as two independent pixel labeling tasks to respectively segment row objects and segment column objects. It finetunes the FCN-2s model [7] that has been pretrained on the PASCAL VOC semantic segmentation challenge [8] for natural scenes. While it only requires an image as input, it does not perform as well as heuristic methods on PDFs. This may be because both rows (columns) and non-row (non-column) regions often contain large amounts of white background (e.g. space between cells in a row), which makes the pixel classes difficult to discriminate from local information. In contrast, SPLERGE pools information over large regions to use more global information. Clinchat et al. [9] compare two learning models on scanned images of seventeeth century handwritten tables. They focus on this narrow problem and use template matching to recover columns and convert the images into "PDF-like" documents. To recover table rows, they use either a Conditional Random Field (CRF) or their proposed Edge Convolutional Networks, an extension of Graph Convolutional Networks [10].

III. SPLERGE

SPLERGE, our proposed table structure extraction method, is composed of two deep learning models that perform split and merge operations in sequence (see Fig. 1). The Split model takes an input image of a well cropped table and produces the grid structure of the table in the form of row and column separators that span the entire image. Because some tables contain spanning cells, we apply the Merge model to the grid output of the Split model to merge together adjacent grid elements to recover the spanning cells.

A. Split Model

The Split model takes as input an image of any dimension $H \times W$ and produces two 1-D output signals: $r \in [0,1]^H$ and $c \in [0,1]^W$. The output signals r and c represent the probability that each row (column) of pixels is part of a logical table row (column) separator region. The Split model is composed of 3 sub-networks:

1) Shared Fully Convolutional Network (SFCN)
2) Row Projection Network (RPN)
3) Column Projection Network (CPN)

The SFCN computes local image features that are used by both the RPN and CPN. The RPN and CPN then take those local features and further process them to predict row and column separators (r and c respectively).

The SFCN is composed of 3 convolution layers with 7x7 kernels, with the last layer performing a dilated convolution [11] with a dilation factor of 2. Each layer produces 18 feature maps and uses the ReLU activation function.

Dilated convolutions, like pooling, increase the receptive field of the network, but unlike pooling, they preserve the spatial resolution of the input. Preserving the input spatial resolution is important in table structure extraction because many column and row separators are only a few pixels wide. In [6], better results were obtained when resizing the initial input to make the separator regions bigger. It is also crucial to have a large receptive field because determining the locations of row and column separators may require global context. For example, text that is consistently left-justified to the same position is indicative of a column separator.

The output of the SFCN is fed as input to both the RPN and CPN. The output of the RPN is r, the probability

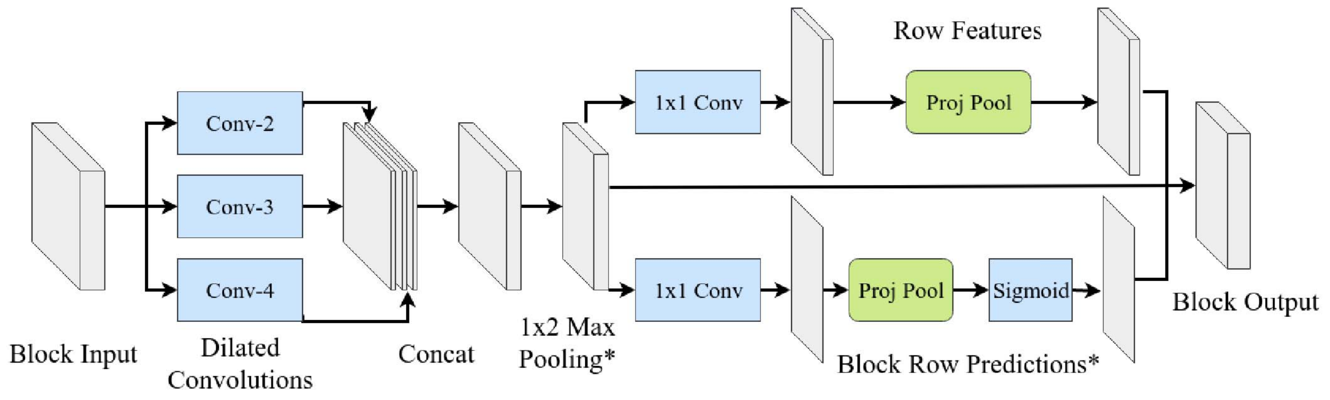


Figure 2. A single block of the Row Projection Network (RPN), which is a sub-network of the Split model. Components marked with * indicate that they are not included in all 5 blocks that compose the RPN. Blocks in the Column Projection Network (CPN) use 2x1 Max Pooling.

that each row of pixels is part of a row separator region. Similarly, the output of the CPN is c. Because the RPN and CPN have identical structure, except for whether projection and pooling operations are over rows or columns of pixels, we focus our discussion only on the RPN.

The RPN in this work is composed of 5 blocks chained together, though any number of blocks can be used. Empirically, using more than 5 blocks did not improve our results, and a similar process was used to determine the other specific architectural choices. To simplify the discussion and notation, we use actual hyperparameter values for our experiments, but other reasonable values could be used. Changing the hyperparameters in a reasonable range did not seem to significantly impact the results in our informal experiments. Fig. 2 shows the operations performed by a single block. First, the input is (in parallel) fed to 3 convolution layers, which have dilation factors of 2/3/4, to produce 6 feature maps each. The output of each dilated convolution is concatenated to obtain 18 feature maps. Using multiple dilation factors allows the RPN to learn multi-scale features and increase its receptive field while still examining more local evidence.

Next, the RPN performs 1x2 max pooling (CPN does 2x1 max pooling). This decreases the width of the feature maps, but maintains the height, so that the output signal r is of size H. Only the first three blocks perform max pooling to make sure that the width is not downsampled too much.

Afterwards, the RPN computes row features (top branch of Fig. 2) through a 1x1 convolution operation followed by *projection pooling* (Fig. 3). Projection pooling is inspired by the projection profile operation used to find gaps of whitespace in classical layout analysis. With projection pooling, we maintain the spatial size of the input (instead of collapsing to 1D as in projection profiles), and simply replace each value in the input with its row average. Specifically,

$$\hat{F}_{ij} = \frac{1}{W} \sum_{j'=1}^{W} F_{ij'} \tag{1}$$

where i indexes over rows in feature map F, and $1 \leq j \leq W$ indexes over columns of F. We call $\hat{F}$ the row projection pooling of F, and apply this operation independently on each feature map, as is typical of pooling operations. Pooling in this fashion allows information to propagate across the entire width of the image, which may be more than 1000 pixels. These row features are concatenated to the output of the max pooling operation, so that each pixel has both local and row-global features. Note that the CPN performs *column* projection pooling, which similarly is

$$\hat{F}_{ij} = \frac{1}{H} \sum_{i'=1}^{H} F_{i'j} \tag{2}$$

The bottom branch of Fig. 2 shows how blocks produce row predictions, though not every block does so. A 1x1 convolution produces a single output map, over which we perform projection pooling. We then apply a sigmoid function to produce probabilities. Since each row of pixels contains a single unique probability, we can take a vertical slice to obtain a 1D signal of probabilities, r^n, where n indicates the block index. To make the intermediate prediction r^n available to block $n+1$, we also concatenate the probabilities (in 2D) to the output of the block.

In our implementation, only the last 3 blocks produce outputs, i.e. r^3, r^4, r^5. During training, we apply a loss to all three predictions, but after training, we only use the last prediction, r^5, for inference. This iterative prediction procedure allows the model to make a prediction and then refine that prediction. Such techniques have been successfully applied in previous work on structured keypoint detection tasks in natural scenes (e.g., human pose [12]).

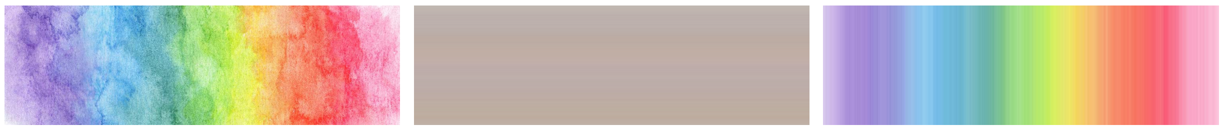
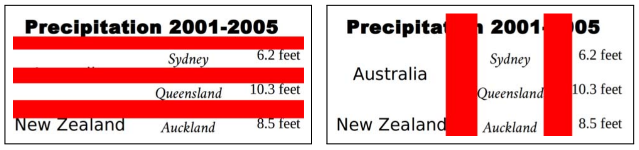

(a) Example Features (b) Row Projection Pooled Features (c) Column Projection Pooled Features

Figure 3. Illustration of projection pooling operation. Every pixel in (a) is replace by its row average (b) or its column average (c).



(a) Row (b) Column

Figure 4. Example 1D GT row and column signals for training the Split model. The 1D signals are replicated to 2D and super imposed on the input image for viewing purposes.

1) Training: The SFCN, RPN, and CPN sub-networks are jointly trained in a typical supervised fashion on table images at 150 DPI. We assume that the images are cropped to only include the table cells and exclude table titles, captions, and footnotes that are not in the cell region.

Each table has annotated ground truth (GT) 1D signals r^* and c^*. The GT is designed to maximize the size of the separator regions without intersecting any non-spanning cell content, as shown in Fig. 4. This is contrary to the traditional notion of cell separators, which for many tables are thin lines that are only a few pixels thick. Predicting small regions is more difficult than large regions, and in the case of unlined tables, the exact location of the cell separator is ill defined. The GT separator regions may intersect cell content for cells that span multiple rows or columns. The goal of the Split model is to recover the basic grid of the table, and spanning cells are handled by the Merge model.

The loss function is the average element-wise binary cross entropy between the block predictions and the GT signals:

$$L(r, r^*) = \frac{1}{H} \sum_{i=1}^{H} r_i^* \log r_i + (1 - r_i^*) \log(1 - r_i) \qquad (3)$$

To prevent overfitting, we modify equation 3 to clip the per-element loss to 0 when $|r_i^* - r_i| < 0.1$. The total loss is a weighted sum of individual output losses:

$$\begin{aligned} L_{tot} &= L(r^5, r^*) + \lambda_4 L(r^4, r^*) + \lambda_3 L(r^3, r^*) \\ &+ L(c^5, c^*) + \lambda_4 L(c^4, c^*) + \lambda_3 L(c^3, c^*) \end{aligned} \qquad (4)$$

where we set $\lambda_4 = 0.25$ and $\lambda_3 = 0.1$.

We train the model from random initialization with the ADAM optimizer [13] for approximately 10^6 weight updates. We use a batch size of 1 because the table images have different spatial sizes. We use an initial learning rate of 0.00075 and decay it by a factor of 0.75 every 80K updates.

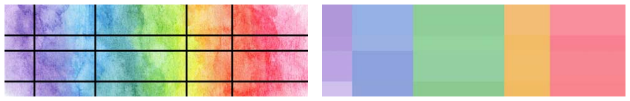

(a) Features with Grid Structure (b) Grid Pooled Features

Figure 5. Illustration of grid pooling using the black grid structure visualized on top of the features in (a). Values are averaged within each grid cell to produce (b).

2) Inference: Once we have predicted r, we need to infer at which pixel positions the row separators occur. For simplicity, the discussion focuses on r, but the same procedure applies to c to obtain column separators. To do so, we partition the image into rows and row separator regions by performing a graphcut segmentation [14] over r. Then we choose the row pixel positions corresponding to the midpoints of each inferred separator region.

To create the graph for segmenting r, we have H nodes arranged in a linear chain, where each node is connected to its two neighbors (except the two nodes at either end). The neighbor edge weights are uniformly set to $\lambda_{gc} = 0.75$. Node i is connected to the source node with an edge weight of r_i and to the sink node with an edge weight of $1 - r_i$.

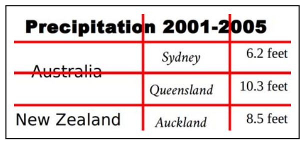
B. Merge Model

The Merge model uses the input image and the outputs of the Split model to predict which grid elements need to be merged to recover cells that span multiple rows or columns. The input tensor is a concatenation of the table image, the output row/col probabilities (r, c), the inferred row/col regions, and the predicted grid structure. The predicted probabilities, r and c, are transformed to a 2D image by stacking (i.e., $[r, r, \ldots, r]$). The inferred row/col regions are rendered as binary mask (similar to the red regions in Fig. 4). The predicted grid structure is rendered as a binary mask where the midpoint of each row and column separator region is rendered as a 7-pixel wide line. Additionally, the grid structure is used to determine the pooling regions of the model.

If the grid structure is composed of M rows and N columns, then the model outputs two matrices:

1) D - probs. of up-down merges (size $(M-1) \times N$)
2) R - probs. of left-right merges (size $M \times (N-1)$)

D_{ij} is the probability of merging cell (i, j) with cell $(i+1, j)$, and R_{ij} is the probability of merging cells (i, j) and $(i, j+1)$. D is not of size $M \times N$ because there are only $M - 1$ pairs of up-down merges in any column. In our



Precipitation 2001-2005		
Australia	Sydney	6.2 feet
	Queensland	10.3 feet
New Zealand	Auckland	8.5 feet

(a) Image with Grid Structure

$$\begin{matrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{matrix}$$

(b) GT D

$$\begin{matrix} 1 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{matrix}$$

(c) GT R

Figure 6. Example GT for training the Merge model for the structure predicted by the Split model.

formulation, all of these probabilities are independent, i.e., a single grid element may be merged in multiple directions.

The Merge model architecture is similar to the Split model. There is a set of 4 shared convolution layers (no dilation), where 2x2 average pooling occurs after layers 2 and 4. Afterwards, the model has 4 branches, where each branch predicts an $M \times N$ matrix of probabilities that cells should merge in a particular direction, i.e. up, down, left, or right. Call these matrices u, d, l, r. While our independence assumption suggests that we multiply the two individual probabilities together in Eqs. 5, 6, this would introduce optimization difficulties when both probabilities are near 0, so we compute D and R as

$$D_{ij} = \frac{1}{2}u_{i+1,j}d_{ij} + \frac{1}{4}(u_{i+1,j} + d_{ij}) \tag{5}$$

$$R_{ij} = \frac{1}{2}l_{i,j+1}r_{ij} + \frac{1}{4}(l_{i,j+1} + r_{ij}) \tag{6}$$

Intuitively, we only predict that a pair of cells should be merged if there is consistency between the individual branch outputs.

Each branch is composed of 3 blocks that are similar to the Split model block shown in Fig. 2. The differences are that the parallel convolution layers use 1/2/3 dilation factors, no max pooling is performed, and projection pooling is replaced with *grid pooling* (Fig. 5). In grid pooling, every pixel location replaces its value with the average of all pixels within its grid element:

$$\hat{F}_{ij} = \frac{1}{|\Omega(i,j)|} \sum_{i',j' \in \Omega(i,j)} F_{i'j'} \tag{7}$$

where $\Omega(i,j)$ is the set of coordinates for all pixels that share the same grid element as (i,j). After grid pooling, all pixels inside the same grid element share the same value, which allows information to propagate within each cell. Subsequent convolutions allow information to propagate between neighboring cells. To produce the u, d, l, or r matrix for a given branch, we average the predicted per-pixel probabilities in each grid element and arrange them in an $M \times N$ matrix. Like the Split model, the Merge model also performs iterative output refinement, where blocks 2 and 3 produce output predictions.

1) Training: Because the Split and Merge models are intended to be used in sequence, we train the Merge model using the grid structures produced by the Split model. To construct the GT D and R matrices (see Fig. 6), we

1) Iterate over all spanning cells in the table
2) Identify which grid elements intersect the GT bounding box of the spanning cell
3) Set the probability of merge for each of these cells to 1 for the appropriate directions

As in the Split model, the loss function for each output is the average (clipped) element-wise binary cross entropy (Eq. 4). The total loss is then

$$\begin{aligned} L_{tot} &= L(D^3, D^*) + \lambda_2 L(D^2, D^*) \\ &+ L(R^3, R^*) + \lambda_2 L(R^2, R^*) \end{aligned} \tag{8}$$

Because spanning cells only occur in 15% of tables in the private dataset used to train our models, we subsample this set so that 50% of the training set for the Merge model has at least one pair of cells that require merging. The training hyperparameters are similar to those of the Split model.

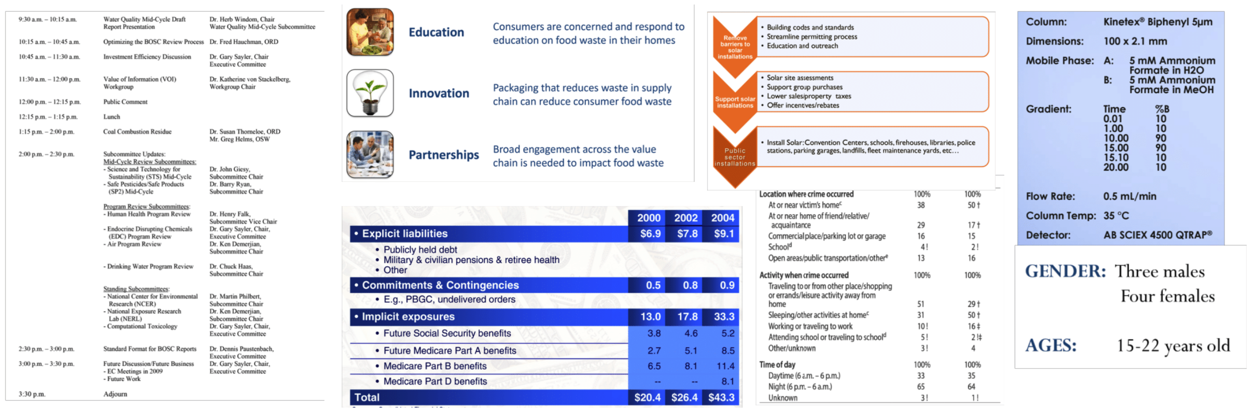
Inference is performed by thresholding D and R at 0.5 probability, and merging the indicated cells. The network prediction has no constraint that the resulting merges produce only rectangular cells, so in post-processing additional merges are added to ensure that the resulting table structure has only rectangular cells. For example, 3 grid elements merged together to form an L-shaped cell would be merged with a 4th element to create a cell that spans 2 rows and 2 columns.

IV. Experimental Results

We tested SPLERGE on two datasets, including the ICDAR 2013 Table Competition benchmark dataset [2], where we achieve state-of-the-art results. This dataset is composed of 156 tables with unambiguous structure that were extracted from PDFs downloaded from government websites. The entire set of PDF tables is considered test data.

The second dataset is a private collection of 93,000 tables taken from web-scraped PDFs that have been manually annotated with structure information. These tables exhibit a larger variety of complexity, visual appearance, and structure than the ICDAR 2013 dataset. For example, there are only 2 completely unlined tables in ICDAR 2013, but the private collection is 15% unlined tables Unlike the ICDAR 2013 dataset, the private collection includes unlined tables with multi-line cells. It also contains tables with poorly aligned columns, cells with complex structure, raster images inside of cells, and complex headers (such as nested headers and headers in the middle of the table). Some example tables are shown in Fig. 7.

We randomly split this dataset into 83K/5K/5K for training, validation, and testing. Because tables from the same PDF exhibit some similarity, we ensured that the set of PDFs used to construct each data split are disjoint.



9:30 a.m. – 10:15 a.m.	Water Quality Mid-Cycle Draft Report Presentation	Dr. Herb Windom, Chair Water Quality Mid-Cycle Subcommittee
10:15 a.m. – 10:45 a.m.	Optimizing the BOSC Review Process	Dr. Fred Hauchman, ORD
10:45 a.m. – 11:30 a.m.	Investment Efficiency Discussion	Dr. Gary Sayler, Chair Executive Committee
11:30 a.m. – 12:00 p.m.	Value of Information (VOI) Workgroup	Dr. Katherine von Stackelberg, Workgroup Chair
12:00 p.m. – 12:15 p.m.	Public Comment	
12:15 p.m. – 1:15 p.m.	Lunch	
1:15 p.m. – 2:00 p.m.	Coal Combustion Residue	Dr. Susan Thorneloe, ORD Mr. Greg Helms, OSW
2:00 p.m. – 2:30 p.m.	Subcommittee Updates:	
	Mid-Cycle Review Subcommittees:	
	- Science and Technology for Sustainability (STS) Mid-Cycle	Dr. John Giesy, Subcommittee Chair
	- Safe Pesticides/Safe Products (SP2) Mid-Cycle	Dr. Barry Ryan, Subcommittee Chair
	Program Review Subcommittees:	
	- Human Health Program Review	Dr. Henry Falk, Subcommittee Vice Chair
	- Endocrine Disrupting Chemicals (EDC) Program Review	Dr. Gary Sayler, Chair, Executive Committee
	- Air Program Review	Dr. Ken Demerjian, Subcommittee Chair
	- Drinking Water Program Review	Dr. Chuck Haas, Subcommittee Chair
	Standing Subcommittees:	
	- National Center for Environmental Research (NCER)	Dr. Martin Philbert, Subcommittee Chair
	- National Exposure Research Lab (NERL)	Dr. Ken Demerjian, Subcommittee Chair
	- Computational Toxicology	Dr. Gary Sayler, Chair, Executive Committee
2:30 p.m. – 3:00 p.m.	Standard Format for BOSC Reports	Dr. Dennis Paustenbach, Executive Committee
3:00 p.m. – 3:30 p.m.	Future Discussion/Future Business - EC Meetings in 2009 - Future Work	Dr. Gary Sayler, Chair, Executive Committee
3:30 p.m.	Adjourn	

Education	Consumers are concerned and respond to education on food waste in their homes
Innovation	Packaging that reduces waste in supply chain can reduce consumer food waste
Partnerships	Broad engagement across the value chain is needed to impact food waste

	2000	2002	2004
• Explicit liabilities	$6.9	$7.8	$9.1
• Publicly held debt • Military & civilian pensions & retiree health • Other			
• Commitments & Contingencies	0.5	0.8	0.9
• E.g., PBGC, undelivered orders			
• Implicit exposures	13.0	17.8	33.3
• Future Social Security benefits	3.8	4.6	5.2
• Future Medicare Part A benefits	2.7	5.1	8.5
• Medicare Part B benefits	6.5	8.1	11.4
• Medicare Part D benefits	--	--	8.1
Total	$20.4	$26.4	$43.3

Remove barriers to solar installations	• Building codes and standards • Streamline permitting process • Education and outreach
Support solar installations	• Solar site assessments • Support group purchases • Lower sales/property taxes • Offer incentives/rebates
Public sector installations	• Install Solar: Convention Centers, schools, firehouses, libraries, police stations, parking garages, landfills, fleet maintenance yards, etc…

Location where crime occurred	100%	100%
At or near victim's home[c]	38	50 †
At or near home of friend/relative/ acquaintance	29	17 †
Commercial place/parking lot or garage	16	15
School[d]	4 !	2 !
Open areas/public transportation/other[e]	13	16
Activity when crime occurred	100%	100%
Traveling to or from other place/shopping or errands/leisure activity away from home	51	29 †
Sleeping/other activities at home[c]	31	50 †
Working or traveling to work	10 !	16 ‡
Attending school or traveling to school[d]	5 !	2 !‡
Other/unknown	3 !	4
Time of day	100%	100%
Daytime (6 a.m. – 6 p.m.)	33	35
Night (6 p.m. – 6 a.m.)	65	64
Unknown	3 !	1 !

Column:	Kinetex® Biphenyl 5µm	
Dimensions:	100 x 2.1 mm	
Mobile Phase:	A:	5 mM Ammonium Formate in H2O
	B:	5 mM Ammonium Formate in MeOH
Gradient:	Time	%B
	0.01	10
	1.00	10
	10.00	90
	15.00	90
	15.10	10
	20.00	10
Flow Rate:	0.5 mL/min	
Column Temp:	35 °C	
Detector:	AB SCIEX 4500 QTRAP®	

GENDER:	Three males Four females
AGES:	15-22 years old

Figure 7. Example table images from the private collection. Some tables in the private collection have complex cells, such as multi-line text (in white space tables), nested tables, vector graphics, and raster images.

Table I
ICDAR 2013 TABLE DATASET RESULTS.

Category	Method	F-measure	Recall	Precision	PDF Input	Image Input
Proposed Models	Split	86.79	86.64	86.93		✓
	Split-PDF	91.63	91.26	92.00	✓	✓
	SPLERGE-PDF	90.89	90.44	91.36	✓	✓
	SPLERGE-PDF + Heuristics	91.95	91.46	92.45	✓	✓
	Split + Heuristics	93.00	92.24	93.78	✓	✓
	Split-PDF + Heuristics	**95.26**	**94.64**	**95.89**	✓	✓
Previously Reported Results	Nurminen [3]	94.60	94.09	95.12	✓	✓
	TEXUS [4]	82.59	84.23	81.02	✓	
	Shigarov [5] C_1	91.50	91.21	91.80	✓	
	Shigarov [5] C_2	93.64	92.33	94.99	✓	
	DeepDeSRT [6]	91.44†	87.36†	95.93†		✓

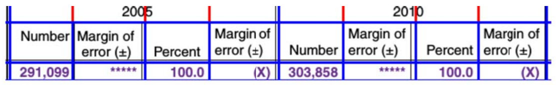
† result not directly comparable due to evaluation on a random subset of 34 tables.

We also experimented with giving the models additional input channels, though the model does not require this. If the input table images are rendered from PDFs, then by examining drawing command metadata we can render additional images that contain only text, only vector art (e.g., lines), or only raster images. These additional images can be concatenated with the rendered PDF and slightly improve network performance.

A. ICDAR 2013

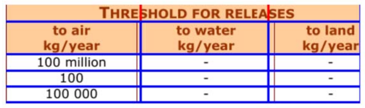
Our results on the ICDAR 2013 dataset are from models trained on the private collection. We attempted to validate that our improved performance results from a better deep model and not just from a larger training set. We did so by reimplementing the DeepDeSRT model [6] and training on the same data as our proposed model.

Table I shows the results on the ICDAR 2013 Table Competition dataset (task 2). Our methods with the *-PDF* suffix indicate that the additional PDF rendered input channels were used. The evaluation metric for this dataset is the F-measure over detected *adjacency relationships* [15]. Roughly speaking, this measures the percentage of correctly detected pairs of adjacent cells, where both cells are segmented correctly and identified as neighbors.



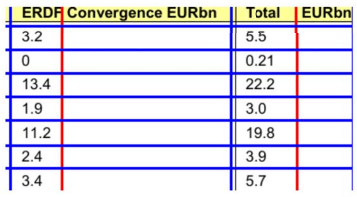
2005				2010			
Number	Margin of error (±)	Percent	Margin of error (±)	Number	Margin of error (±)	Percent	Margin of error (±)
291,099	*****	100.0	(X)	303,858	*****	100.0	(X)

(a) Blank Cells next to Headers



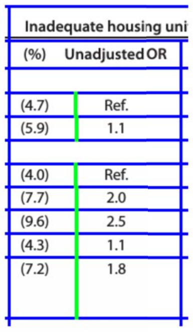
THRESHOLD FOR RELEASES		
to air kg/year	to water kg/year	to land kg/year
100 million	-	-
100	-	-
100 000	-	-

(b) Separators Through Text



ERDF Convergence EURbn	Total EURbn
3.2	5.5
0	0.21
13.4	22.2
1.9	3.0
11.2	19.8
2.4	3.9
3.4	5.7

(c) Join Columns

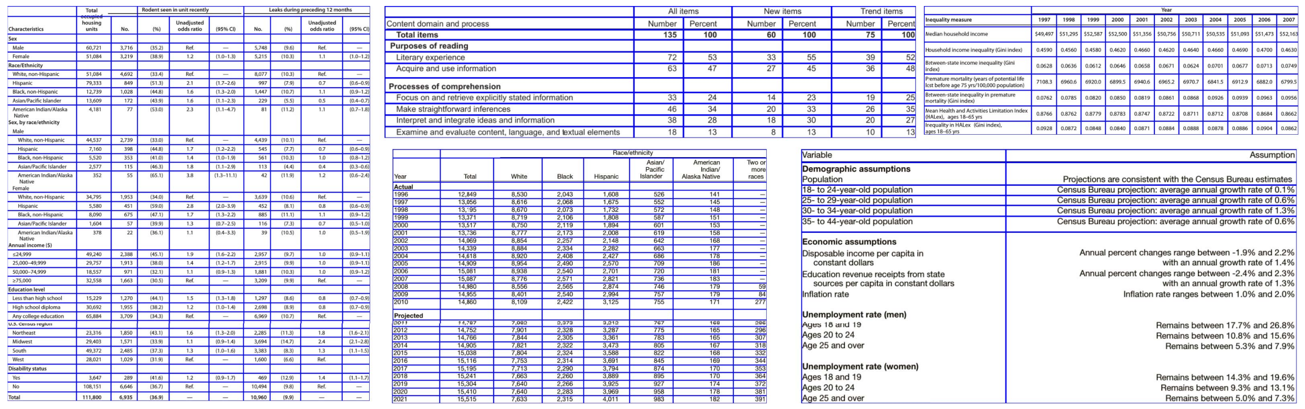


Inadequate housing uni	
(%)	Unadjusted OR
(4.7)	Ref.
(5.9)	1.1
(4.0)	Ref.
(7.7)	2.0
(9.6)	2.5
(4.3)	1.1
(7.2)	1.8

(d) Split Columns

Figure 8. Example heuristic post-processing on ICDAR2013 table segments. Removed separators are shown in red, added separators in green, and unmodified predicted separators in blue.

For this dataset, the Merge model failed to provide adequate post-processing for the output of the Split model. After performing the predicted merges, our post processing combines additional cells to prevent cells from having an



	Total occupied housing units	Rodent seen in unit recently				Leaks during preceding 12 months			
Characteristics		No.	(%)	Unadjusted odds ratio	(95% CI)	No.	(%)	Unadjusted odds ratio	(95% CI)
Sex									
Male	60,721	3,716	(35.2)	Ref.	—	5,748	(9.6)	Ref.	—
Female	51,084	3,219	(38.9)	1.2	(1.0–1.3)	5,215	(10.3)	1.1	(1.0–1.2)
Race/Ethnicity									
White, non-Hispanic	51,084	4,692	(33.4)	Ref.	—	8,077	(10.3)	Ref.	—
Hispanic	79,333	849	(51.3)	2.1	(1.7–2.6)	997	(7.9)	0.7	(0.6–0.9)
Black, non-Hispanic	12,739	1,028	(44.8)	1.6	(1.3–2.0)	1,447	(10.7)	1.1	(0.9–1.2)
Asian/Pacific Islander	13,609	172	(43.9)	1.6	(1.1–2.3)	229	(5.5)	0.5	(0.4–0.7)
American Indian/Alaska Native Sex, by race/ethnicity Male	4,181	77	(53.0)	2.3	(1.1–4.7)	81	(11.2)	1.1	(0.7–1.8)
White, non-Hispanic	44,537	2,739	(33.0)	Ref.	—	4,439	(10.1)	Ref.	—
Hispanic	7,160	398	(44.8)	1.7	(1.2–2.2)	545	(7.7)	0.7	(0.6–0.9)
Black, non-Hispanic	5,520	353	(41.0)	1.4	(1.0–1.9)	561	(10.3)	1.0	(0.8–1.2)
Asian/Pacific Islander	2,577	115	(46.3)	1.8	(1.1–2.9)	113	(4.4)	0.4	(0.3–0.6)
American Indian/Alaska Native Female	352	55	(65.1)	3.8	(1.3–11.1)	42	(11.9)	1.2	(0.6–2.4)
White, non-Hispanic	34,795	1,953	(34.0)	Ref.	—	3,639	(10.6)	Ref.	—
Hispanic	5,580	451	(59.0)	2.8	(2.0–3.9)	452	(8.1)	0.8	(0.6–0.9)
Black, non-Hispanic	8,090	675	(47.1)	1.7	(1.3–2.2)	885	(11.1)	1.1	(0.9–1.2)
Asian/Pacific Islander	1,604	57	(39.9)	1.3	(0.7–2.5)	116	(7.3)	0.7	(0.5–1.0)
American Indian/Alaska Native Annual income ($)	378	22	(36.1)	1.1	(0.4–3.3)	39	(10.5)	1.0	(0.5–1.9)
≤24,999	49,240	2,388	(45.1)	1.9	(1.6–2.2)	2,957	(9.7)	1.0	(0.9–1.1)
25,000–49,999	29,757	1,913	(38.0)	1.4	(1.2–1.7)	2,915	(9.9)	1.0	(0.9–1.1)
50,000–74,999	18,557	971	(32.1)	1.1	(0.9–1.3)	1,881	(10.3)	1.0	(0.9–1.2)
≥75,000	32,558	1,663	(30.5)	Ref.	—	3,209	(9.9)	Ref.	—
Education level									
Less than high school	15,229	1,270	(44.1)	1.5	(1.3–1.8)	1,297	(8.6)	0.8	(0.7–0.9)
High school diploma	30,692	1,955	(38.2)	1.2	(1.0–1.4)	2,698	(8.9)	0.8	(0.7–0.9)
Any college education	65,884	3,709	(34.3)	Ref.	—	6,969	(10.7)	Ref.	—
U.S. Census region									
Northeast	23,316	1,850	(43.1)	1.6	(1.3–2.0)	2,285	(11.3)	1.8	(1.6–2.1)
Midwest	29,403	1,571	(33.9)	1.1	(0.9–1.4)	3,694	(14.7)	2.4	(2.1–2.8)
South	49,372	2,485	(37.3)	1.3	(1.0–1.6)	3,383	(8.3)	1.3	(1.1–1.5)
West	28,021	1,029	(31.9)	Ref.	—	1,600	(6.6)	Ref.	
Disability status									
Yes	3,647	289	(41.6)	1.2	(0.9–1.7)	469	(12.9)	1.4	(1.1–1.7)
No	108,151	6,646	(36.7)	Ref.	—	10,494	(9.8)	Ref.	—
Total	111,800	6,935	(36.9)	—	—	10,960	(9.9)	—	—

	All items		New items		Trend items	
Content domain and process	Number	Percent	Number	Percent	Number	Percent
Total items	**135**	**100**	**60**	**100**	**75**	**100**
Purposes of reading						
Literary experience	72	53	33	55	39	52
Acquire and use information	63	47	27	45	36	48
Processes of comprehension						
Focus on and retrieve explicitly stated information	33	24	14	23	19	25
Make straightforward inferences	46	34	20	33	26	35
Interpret and integrate ideas and information	38	28	18	30	20	27
Examine and evaluate content, language, and textual elements	18	13	8	13	10	13

	Year										
Inequality measure	1997	1998	1999	2000	2001	2002	2003	2004	2005	2006	2007
Median household income	$49,497	$51,295	$52,587	$52,500	$51,356	$50,756	$50,711	$50,535	$51,093	$51,473	$52,163
Household income inequality (Gini index)	0.4590	0.4560	0.4580	0.4620	0.4660	0.4620	0.4640	0.4660	0.4690	0.4700	0.4630
Between-state income inequality (Gini index)	0.0628	0.0636	0.0612	0.0646	0.0658	0.0671	0.0624	0.0701	0.0677	0.0713	0.0749
Premature mortality (years of potential life lost before age 75 yrs/100,000 population)	7108.3	6960.6	6920.0	6899.5	6940.6	6965.2	6970.7	6841.5	6912.9	6882.0	6799.5
Between-state inequality in premature mortality (Gini index)	0.0762	0.0785	0.0820	0.0850	0.0819	0.0861	0.0868	0.0926	0.0939	0.0963	0.0956
Mean Health and Activities Limitation Index (HALex), ages 18–65 yrs	0.8766	0.8762	0.8779	0.8783	0.8747	0.8722	0.8711	0.8712	0.8708	0.8684	0.8662
Inequality in HALex (Gini index), ages 18–65 yrs	0.0928	0.0872	0.0848	0.0840	0.0871	0.0884	0.0888	0.0878	0.0886	0.0904	0.0862

	Race/ethnicity						
Year	Total	White	Black	Hispanic	Asian/ Pacific Islander	American Indian/ Alaska Native	Two or more races
Actual							
1996	12,849	8,530	2,043	1,608	526	141	—
1997	13,056	8,616	2,068	1,675	552	145	—
1998	13,195	8,670	2,073	1,732	572	148	—
1999	13,371	8,719	2,106	1,808	587	151	—
2000	13,517	8,750	2,119	1,894	601	153	—
2001	13,736	8,777	2,173	2,008	619	158	—
2002	14,069	8,854	2,257	2,148	642	168	—
2003	14,339	8,884	2,334	2,282	663	177	—
2004	14,618	8,920	2,408	2,427	686	178	—
2005	14,909	8,954	2,490	2,570	709	186	—
2006	15,081	8,938	2,540	2,701	720	181	—
2007	15,087	8,776	2,571	2,821	736	183	—
2008	14,980	8,556	2,565	2,874	746	179	59
2009	14,955	8,401	2,540	2,994	757	179	84
2010	14,860	8,109	2,422	3,125	755	171	277
Projected							
2011	14,787	7,082	2,373	3,213	767	168	286
2012	14,752	7,901	2,328	3,287	775	165	296
2013	14,766	7,844	2,305	3,361	783	165	307
2014	14,905	7,821	2,322	3,473	805	167	318
2015	15,038	7,804	2,324	3,588	822	168	332
2016	15,116	7,753	2,314	3,691	845	169	344
2017	15,195	7,713	2,290	3,794	874	170	353
2018	15,241	7,663	2,260	3,889	895	170	364
2019	15,304	7,640	2,266	3,925	927	174	372
2020	15,410	7,640	2,283	3,969	958	178	381
2021	15,515	7,633	2,315	4,011	983	182	391

Variable	Assumption
Demographic assumptions Population	Projections are consistent with the Census Bureau estimates
18- to 24-year-old population	Census Bureau projection: average annual growth rate of 0.1%
25- to 29-year-old population	Census Bureau projection: average annual growth rate of 0.6%
30- to 34-year-old population	Census Bureau projection: average annual growth rate of 1.3%
35- to 44-year-old population	Census Bureau projection: average annual growth rate of 0.6%
Economic assumptions Disposable income per capita in constant dollars Education revenue receipts from state sources per capita in constant dollars Inflation rate **Unemployment rate (men)** Ages 18 and 19 Ages 20 to 24 Age 25 and over **Unemployment rate (women)** Ages 18 and 19 Ages 20 to 24 Age 25 and over	Annual percent changes range between -1.9% and 2.2% with an annual growth rate of 1.4% Annual percent changes range between -2.4% and 2.3% with an annual growth rate of 1.3% Inflation rate ranges between 1.0% and 2.0% Remains between 17.7% and 26.8% Remains between 10.8% and 15.6% Remains between 5.3% and 7.9% Remains between 14.3% and 19.6% Remains between 9.3% and 13.1% Remains between 5.0% and 7.3%

Figure 9. Example predictions on challenging (mostly unlined) ICDAR 2013 tables, including one failure case.

Table II
PRIVATE COLLECTION RESULTS.

Category	Method	Table Accuracy	F-measure	Recall	Precision	PDF Input	Image Input
Proposed Models	Split	80.79	95.46	**95.87**	95.06		✓
	SPLERGE	83.09	95.78	95.50	96.07		✓
	Split-PDF	81.07	95.37	95.64	95.10	✓	✓
	SPLERGE-PDF	**85.65**	**95.92**	95.59	**96.27**	✓	✓
Our Implementation	DeepDeSRT	51.76	76.85	75.32	78.44		✓
	DeepDeSRT-PDF	53.48	78.44	77.03	79.91	✓	✓
Commercial	Acrobat	61.11	82.62	82.66	82.59	✓	

L-shape in the final output. In several of the large header regions in the ICDAR 2013 dataset, large groups of individual cells were merged into a single cell due to a few incorrect pairwise merge predictions that created L-shapes. Instead of further refining the heuristic to prevent L-shapes, we instead implemented some simple heuristics that could replace the Merge model. These heuristics include

- Merge cells where predicted separators pass through text.
- Merge adjacent columns when the vast majority of paired cells (after row 3) are either both blank or only one cell per pair is non-blank. This merges a content column with a (mostly) blank column.
- In the first row (likely header row), merge non-blanks cells with adjacent blank cells.
- Split columns that have a consistent whitespace gap between vertically aligned text.

Some example tables that are fixed by the heuristics are shown in Fig. 8. While the Split model performs well in identifying the table grid, it sometimes makes easily correctable mistakes and cannot by itself handle spanning cells. When combined with simple heuristics to handle these cases, it achieves an F-measure of 95.26% compared to the previous best result of 94.60%. The Merge model fails to generalize from the private collection to the ICDAR 2013 dataset, but as shown in Table II, it does improve performance on the private collection. Figure 9 shows some example predictions by *Split-PDF + Heuristics* for unlined tables, which are harder to recognize than lined tables.

There is a large performance difference from when PDF information (text, path, image channels) is given as input to the Split model and when it is not. Because the difference is not as large on the private collection (Table II), we conclude that the usefulness of PDF input channels depends on the dataset. The ICDAR tables are primarily lined, have larger headers, and may have distinct visual appearances compared to the training dataset. Thus, the additional PDF information may help more in the unfamiliar domain because the text and path elements are explicit inputs instead of needing to be visually inferred by the model.

We reimplemented the DeepDeSRT table structure model [6] and trained it on the same private data as our proposed model. However, we were unable to obtain reasonable performance, even after exploring a variety of values for post processing thresholds and training hyperparameters (values not specified in [6]). In [6], they report an FM of 91.44%, but over a random subset of 34 tables, so a direct comparison cannot be done. we felt that this gap indicated that we could not faithfully reproduce their model for a fair comparison. However, the training set we used is very different and there are significant differences between the private collection and the ICDAR 2013 dataset. These reasons could explain the performance gap, but the gap is large enough that we are unsure that our implementation is a faithful reproduction of Deep DeSRT, so we omit exact performance numbers to avoid direct comparison.

B. Private Collection

On this dataset, we evaluate methods using precision and recall over correctly detected cells. We also report accuracy as the percentage of tables with perfect precision and recall. A predicted bounding box (BB) is a correct prediction if it entirely contains only a single ground truth cell content BB. In particular, predicted BBs that intersect multiple GT BBs or that do not completely contain any GT BB are marked as false positives. Unmatched GT BBs are marked as false negatives. Because blank cells were not manually annotated, we exclude predicted BBs that do not intersect any GT BBs. This way, methods are not penalized if they correctly predict unmarked blank cells.

Table II shows the results on the test split of 5000 tables. Following [2], reported precision and recall are computed per-table and then averaged. We were unable to find any official implementations of prior works, so for comparison, we use the commercial software system Acrobat Pro DC and our reimplementation of the DeepDeSRT model [6].

All variants of our proposed models significantly outperform the two baselines on all metrics. We also see that the Merge model significantly improves table accuracy, as the Split model alone cannot handle tables with spanning cells. For tables that require merges, the average number of needed merges per-table is much less for the private collection than for ICDAR 2013, resulting in fewer L-shaped predictions. We also observe that using PDF information as input does bring an improvement, but more marginally than with the ICDAR 2013 data. This may reflect the fact that domain differences between ICDAR 2013 and the private collection are more apparent in the rendered PDF, but the differences are lessened when examining just the text or path image channels. This suggests that this method may be effective for scanned documents, where only the captured image is available for input, if appropriate training data is provided.

V. Conclusion

We have proposed a new method for table structure extraction, SPLERGE. It is composed of a pair of deep learning models that together split a table image into the basic grid of cells, and then merge cells together to recover cells that span multiple rows and columns. The key insight of the models is to pool information over large regions of the table image such as entire rows/columns of pixels or previously predicted cell regions. When evaluating the split model on the ICDAR 2013 Table Competition dataset, we achieve state-of-the-art performance. We also show that PDF information, such as if page elements are text/path/image, can be encoded as input to the deep network and improve performance. However, if such information is not available (e.g., scanned documents), the model can use only the grayscale image as input. Lastly, we have shown that the Merge model is effective on a private collection of tables extracted from the web.

References

[1] J. Hu, R. Kashi, D. Lopresti, G. Nagy, and G. Wilfong, "Why table ground-truthing is hard," in *International Conference on Document Analysis and Recognition*. IEEE, 2001, pp. 129–133.

[2] M. Göbel, T. Hassan, E. Oro, and G. Orsi, "ICDAR 2013 table competition," in *International Conference on Document Analysis and Recognition*. IEEE, 2013, pp. 1449–1453.

[3] A. Nurminen, "Algorithmic extraction of data in tables in PDF documents," Master's thesis, Tampere University of Technology, 2013.

[4] R. Rastan, H.-Y. Paik, and J. Shepherd, "TEXUS: A task-based approach for table extraction and understanding," in *Symposium on Document Engineering*. ACM, 2015, pp. 25–34.

[5] A. Shigarov, A. Mikhailov, and A. Altaev, "Configurable table structure recognition in untagged PDF documents," in *Symposium on Document Engineering*. ACM, 2016, pp. 119–122.

[6] S. Schreiber, S. Agne, I. Wolf, A. Dengel, and S. Ahmed, "DeepDeSRT: Deep learning for detection and structure recognition of tables in document images," in *International Conference on Document Analysis and Recognition*, vol. 1. IEEE, 2017, pp. 1162–1167.

[7] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[8] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes (VOC) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010.

[9] S. Clinchant, H. Déjean, J.-L. Meunier, E. M. Lang, and F. Kleber, "Comparing machine learning approaches for table recognition in historical register books," in *International Workshop on Document Analysis Systems*. IEEE, 2018, pp. 133–138.

[10] Q. Xu, Q. Wang, C. Xu, and L. Qu, "Collective vertex classification using recursive neural network," *arXiv preprint arXiv:1701.06751*, 2017.

[11] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015.

[12] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, "Convolutional pose machines," in *Computer Vision and Pattern Recognition*, 2016, pp. 4724–4732.

[13] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[14] D. M. Greig, B. T. Porteous, and A. H. Seheult, "Exact maximum a posteriori estimation for binary images," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 271–279, 1989.

[15] M. Göbel, T. Hassan, E. Oro, and G. Orsi, "A methodology for evaluating algorithms for table understanding in PDF documents," in *Symposium on Document Engineering*. ACM, 2012, pp. 45–48.