Table Detection via Probability Optimization

Yalin Wang[†] Ihsin T. Phillips[‡] Robert M. Haralick[*]

[†]Dept. of Elect. Eng.
Univ. of Washington
Seattle, WA 98195
ylwang@u.washington.edu

[‡]Dept. of Comp. Science
Queens College, CUNY
Flusing, NY 11367
yun@image.cs.qc.edu

[*]The Graduate School
CUNY
New York, NY 10016
haralick@gc.cuny.edu

Abstract

This paper presents a table detection algorithm using optimization method. We define the table detection problem within the whole page segmentation framework. To reach a good table detection result, we emphasize to optimize the probabilities of the table region , its neighboring text block and their separator. An iterative updating method is used to optimize the whole page segmentation probability. The training and testing data set for the algorithm include $1,125$ *document pages having* 518 *table entities and a total of* $10,934$ *cell entities. Compared with our previous work [12], it raised the accuracy rate to* 95.67% *from* 90.32% *and to* 97.05% *from* 92.04%.

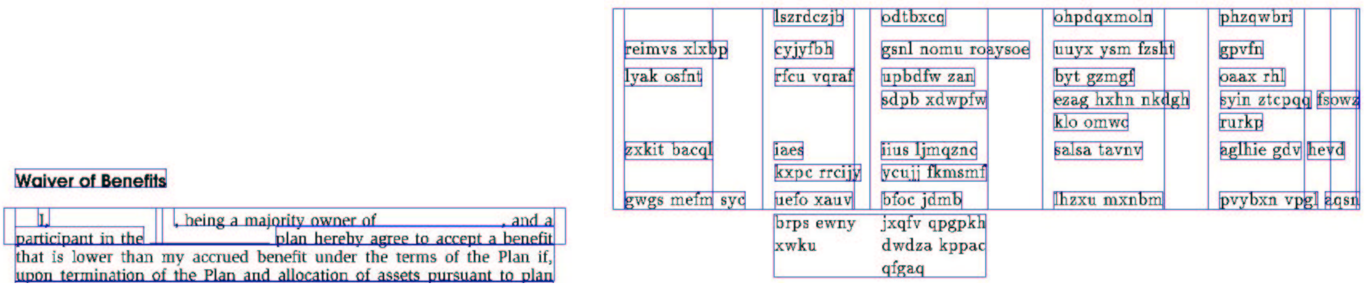
1 Introduction

The large number of existing documents and the production of a multitude of new ones every year raise important issues in efficient handling, retrieval and storage of these documents and the information which they contain. This has led to the emergence of research domains dealing with the recognition of the constituent elements of documents and automatic analyses of the overall physical and logical structures of documents by computers. As a compact and efficient way to present relational information, tables are used frequently in many documents.

In our previous research [12], we built an automatic table ground truth generation system. We detected table structure by background analysis and a statistical validation procedure. In our recent research, we improved its table identification performance by incorporating optimization methods.

In the current table identification research, some of them were based on predefined table layout structure ([1], [9]) or relied on complex heuristics which were based on local analysis([4]). Although an algorithm([8]) is based on machine learning method, the classifier design and feature are only limited in single column or row. A dynamic programming table identification algorithm was given in [3]. It detected tables based on computing an optimal partitioning of a document into some number of tables. Because it is ASCII text based, it cannot fully make use of document image information when applied to document images. [5] presents an optimization algorithm so-called Document Image Decoding(DID) method. It patterned after the use of hidden Markov models in speech recognition. It estimated the original message, given the observed image, by finding the *a posteriori probability* using a Viterbi-like dynamic programming algorithm. Recently, a DID-based algorithm so-called turbo decoding([10]) was an example which implemented DID idea to document layout analysis. However, no experimental result has ever been reported on real images.

Our previous table detection work is a background based, coarse to fine table detection algorithm. It is probability based. However, it only has one step. It determined the table candidates by finding the so-called large horizontal blank blocks [11] and statistically validated if it is a real table entity. Its one-step nature made it difficult to reach a high accuracy detection results. Figure 1 shows its two failed examples. One is a false alarm example and the other is a misdetection example. The goal of our current research is to use optimization method to improve table detection results. We defined the table detection as a probability optimization problem. Not only did we consider measurement probability on table entities, but we computed the probabilities of separators and text blocks. An iterative updating method was used to optimize the page segmentation probability. We improved the accuracy rates in our testing data set.


Waiver of Benefits

I, , being a majority owner of , and a participant in the plan hereby agree to accept a benefit that is lower than my accrued benefit under the terms of the Plan if, upon termination of the Plan and allocation of assets pursuant to plan

(a) (b)

Figure 1. Examples of table detection research of our early research; (a) a false alarm example; (b) a misdetection example.

The rest of the paper is organized as follows. We give the problem statement in Section 2. In Section 6, we present our algorithm details. Experimental results are then reported in Section 4 and we conclude with our future directions in Section 5.

2 Problem Statement

Let $\mathcal{A}$ be a set of block entities. Let $\mathcal{L}$ be a set of content labels, such as table, non-table. Function $f : \mathcal{A} \rightarrow \mathcal{L}$ associates each element of $\mathcal{A}$ with a label. Function $V : \wp(\mathcal{A}) \rightarrow \Lambda$ specifies measurements made on subset of $\mathcal{A}$, where Λ is the measurement space.

The table detection problem can be formulated as follows: *Given initial set $\mathcal{A}$, find a labeling function $f : A \rightarrow L$, that maximizes the probability:*

$$P(V(\tau) : \tau \in \mathcal{A}, f|\mathcal{A}) = P(V(\tau) : \tau \in \mathcal{A}, |f, \mathcal{A})P(f|\mathcal{A}) \tag{1}$$

By making the assumption of conditional independence that when the label $f(\alpha), \alpha \in \mathcal{A}$ is known no knowledge of other labels will alter the probability of $V(\tau)$, we can decompose the probability in

Equation 1 into

$$P(V(\tau) : \tau \in \mathcal{A}|f, \mathcal{A}) = \underbrace{\prod_{\tau \in \mathcal{A}} P(V(\tau)|f, \mathcal{A})}_{(a)} \underbrace{P(f|\mathcal{A})}_{(b)} \tag{2}$$

According to f function values, item(a) in Equation 2 can be computed by applying different measurement functions V_{TAB} and V_{TXT}.

$$P(V(\tau) : \tau \in \mathcal{A}|f, \mathcal{A}) = \prod_{f(\tau)=table}^{\tau \in \mathcal{A}} P(V_{TAB}(\tau)|f, \mathcal{A}) \prod_{f(\tau)=nontable}^{\tau \in \mathcal{A}} P(V_{TXT}(\tau)|f, \mathcal{A})P(f|\mathcal{A}) \tag{3}$$

To compute item(b) in Equation 2, we consider the discontinuity property between neighbors to two entities with different labels. Let $\mathcal{A} = \{A_1, A_2, \cdots, A_M\}$ be the set of document elements extracted from a document page. Each element $A_i \in \mathcal{A}$ is represented by a bounding box (x, y, w, h), where (x, y) is the coordinate of top-left corner, and w and h are the width and height of the bounding box respectively. The spatial relations between two adjacent boxes are shown in Figure 2.

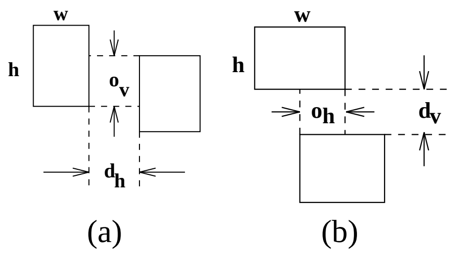


Figure 2. Illustrates the spatial relations between two bounding boxes that are (a) horizontally adjacent (b) vertically adjacent.

For a pair of bounding boxes a and b, the horizontal distance $d_h(a, b)$ and vertical distance $d_v(a, b)$ between them are defined as

$$d_h(a, b) = \begin{cases} x_b - x_a - w_a & \text{if } x_b > x_a + w_a \\ x_a - x_b - w_b & \text{if } x_a > x_b + w_b \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

$$d_v(a, b) = \begin{cases} y_b - y_a - h_a & \text{if } y_b > y_a + h_a \\ y_a - y_b - h_b & \text{if } y_a > y_b + h_b \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

The horizontal overlap $o_h(a, b)$ and vertical overlap $o_v(a, b)$ between a and b are defined as

$$o_h(a, b) = \begin{cases} x_a + w_a - x_b & \text{if } x_b > x_a, x_b < x_a + w_a \\ x_b + w_b - x_a & \text{if } x_a > x_b, x_a < x_b + w_b \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

$$o_v(a, b) = \begin{cases} y_a + h_a - y_b & \text{if } y_b > y_a, y_b < y_a + h_a \\ y_b + h_b - y_a & \text{if } y_a > y_b, y_a < y_b + h_b \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

Let $A_a = (x_a, y_a, w_a, h_a)$ and $A_b = (x_b, y_b, w_b, h_b)$ be two glyphs.

- We define A_b as a *right neighbor* of A_a if $A_b \neq A_a, x_b > x_a$, and $o_v(a,b) > 0$. Let B_a be the set of right neighbors of A_a. A_a and A_b is called *horizontally adjacent* if

$$A_b = \arg\min_{A_i \in B_a} (d_h(a,i) | x_i > x_a, o_v(a,i) > 0). \tag{8}$$

- We define A_b as a *lower neighbor* of A_a if $A_b \neq A_a, y_b > y_a$, and $o_h(a,b) > 0$. Let B_a be the set of right neighbors of A_a. A_a and A_b is called *vertically adjacent* if

$$A_b = \arg\min_{A_i \in B_a} (d_v(a,i) | y_i > y_a, o_h(a,i) > 0). \tag{9}$$

The neighbor set is defined as

$$\mathcal{N} = \{(v_a, v_b) | v_a \text{ and } v_b \text{ horizontally or vertically adjacent}, v_a \in \mathcal{V}, v_b \in \mathcal{V}\}$$

Assume the conditional independence between each neighborhood relationship, $P(f|\mathcal{A})$ can be computed as

$$P(f|\mathcal{A}) = \prod_{\{p,q\} \in \mathcal{N}} P_{\{p,q\}}(f_p, f_q | p, q) \tag{10}$$

where $P_{\{p,q\}}(f_p, f_q|p,q)$ has the property as

$$P_{\{p,q\}}(f_p, f_q|p,q) = \begin{cases} P_{\{p,q\}}(f_p, f_q|p,q) & f_p \neq f_q \\ 0 & f_p = f_q \end{cases} \tag{11}$$

Equation 2 can be further decomposed as

$$P(V(\tau) : \tau \in \mathcal{A} | f, \mathcal{A}) = \prod_{\tau \in \mathcal{A}} P(V(\tau)|f, \mathcal{A}) \prod_{\{p,q\} \in \mathcal{N}} P(f_p, f_q|p,q) \tag{12}$$

Maximizing probability in Equation 1 is equivalent to maxmizing item (a) and item (b) in Equation 2. Our table detection system works as follows: we used our early research [12] to get preliminary table detection results. Then we adjust the labeling by monotonically maximizing the probability until no improvement can be made.

3 Table Detection Algorithm

Figure 5 shows the our algorithm diagram. Given a labelled page, first we will estimate its probability. For each table, we will consider several adjustments, which are to keep it as a table, to grow the table to include its upper/lower neighbor, to merge the table with its upper/lower neighbor and label it as text block. For each table, we compute the new probability under such adjustments. We select the adjustment which produces the biggest improvement upon the initial page segmentation probability. Repeat this process until no improvement can be made. We will give the details to compute the probabilities in the following.

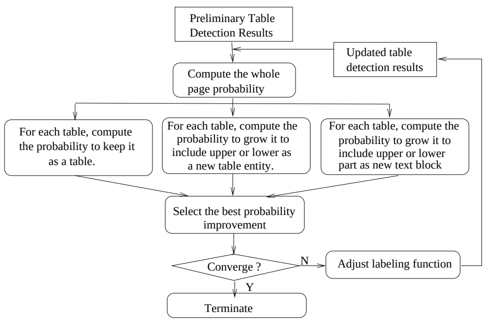

Preliminary Table Detection Results
Updated table detection results
Compute the whole page probability
For each table, compute the probability to keep it as a table.
For each table, compute the probability to grow it to include upper or lower as a new table entity.
For each table, compute the probability to grow it to include upper or lower part as new text block
Select the best probability improvement
Converge ?
N
Adjust labeling function
Y
Terminate

Figure 3. Overview of the table detection algorithm

3.1 Table and Text Separator Probability

Given a table, t and its neighboring text block B, $(t, B) \in \mathcal{N}$. In our current research, we only consider the vertical adjacent neighbors. We compute the probability the space between them being a table and text separator as

$$P(f_t, f_B|t, B) = P(TableTextSeparator|o_h(t, B), d_v(t, B))$$

where the definitions of $d_v(t, B)$ and $o_h(t, B)$ can be found at Equation 5 and Equation 6.

3.2 Table Measurement Probability

To facilitate table detection, we applied our table decomposition algorithm [12] on each detected table. Based on the table decomposition results, three features are computed [12].

- Ratio of total large vertical blank block [11] and large horizontal blank block [11] areas over identified table area. Let t be an identified table and $\mathcal{B}$ be the set of large horizontal horizontal and vertical blank blocks and in it, $ra = \frac{\sum_{\beta \in \mathcal{B}} Area(\beta)}{Area(t)}$;
- Maximum difference of the cell baselines in a row. Denote the set of the cells in a row i as $\mathcal{RC}_i$, $\mathcal{RC}_i = \{c_{i,1}, c_{i,2}, ..., c_{i,i_m}\}$. Denote the set of $\mathcal{RC}_i$ as $\mathcal{RC}$, $\mathcal{RC} = \{\mathcal{RC}_i, i = 1, ..., m\}$, where m is the row number in the table. Let $baseline(c)$ be the y coordinate of the cell entity bottom line, $mc = \max_{\mathcal{RC}_i \in \mathcal{RC}} (\max_{c_{i,j} \in \mathcal{RC}_i} (baseline(c_{i,j})) - \min_{c_{i,j} \in \mathcal{RC}_i} (baseline(c_{i,j})))$;
- Accumulated difference of the justification in all columns. Denote the set of cells in a column, i, in the table $\mathcal{CC}_i = \{c_{i,1}, c_{i,2}, ..., c_{i,i_n}\}$. Denote the set of $\mathcal{CC}_i$ as $\mathcal{CC}$, $\mathcal{CC} = \{\mathcal{CC}_i, i = 1, ..., n\}$, where n is the column number in the table. Let $x_{i,j}, y_{i,j}, w_{i,j}, h_{i,j}$ represent the bounding box of the cell $c_{i,j} \in \mathcal{CC}_i$. We estimate the justification of a column, $i, i = 1, ..., n$, by computing the vertical projection of the left, center, and right edge of $c_{i,j}, j = 1, ..., i_n$,

$$\begin{aligned} C_{left}[i] &= max_{c_{i,j} \in \mathcal{CC}_i}(x_{i,j}) - min_{c_{i,j} \in \mathcal{CC}_i}(x_{i,j}) \\ C_{center}[i] &= max_{c_{i,j} \in \mathcal{CC}_i}(x_{i,j} + w_{i,j}/2) - min_{c_{i,j} \in \mathcal{CC}_i}(x_{i,j} + w_{i,j}/2) \\ C_{right}[i] &= max_{c_{i,j} \in \mathcal{CC}_i}(x_{i,j} + w_{i,j}) - min_{c_{i,j} \in \mathcal{CC}_i}(x_{i,j} + w_{i,j}) \\ J_i &= min\{C_{left}[i], C_{center}[i], C_{right}[i]\} \end{aligned}$$

The accumulated difference of the justification in all columns, mj, is computed as: $mj = \sum_{i=1}^{n} J_i$.

Then we can compute the table consistent probability for table t as

$$P(V_{TAB}(t)) = P(consistence(t)|ra(t), mc(t), mj(t))$$

3.3 Text Block Measurement Probability

A text block usually has homogeneous inter-line spacing and certain alignment type. Given a detected text block B, we compute the probability that B has homogeneous leading, and certain type of text alignment. According to Liang et. al.'s work [7], we compute text block measurement probability as

$$P(V_{TXT}(B)) = P(V_{TXT}(B)|Leading(B), Alignment(B))$$

By making the assumption of conditional independence, we can rewrite the above equation as

$$P(V_{TXT}(B)) = P(V_{TXT}(B)|Leading(B)P(V_{TXT}(B)|Alignment(B))$$

Let $B = (A_1, ..., A_n)$ be an extracted block. $D_B = (d(1), d(2), ..., d(n-1))$ is a sequence of inter-line spaces, where $d(j)$ is the space between A_j and A_{j+1}. We compute the median and the maximum value of the elements of D_B. The probability is

$$P(V_{TXT}(B)|Leading(B)) = P(median(D_B), max(D_B)|Leading(B))$$

.

Given a text block B that consists of a group of text lines $B = (A_1, A_2, \cdots, A_n)$, we determine the text alignment of B by observing the alignment of the text line edges. Let e_{li} be the left edge of the text line A_i and let e_{ci} and e_{ri} be the center and right edges of the line box respectively. Let E_l be the left edges of text line 2 to n, such that $E_l = \{e_{li}|2 \leq i \leq n\}$. E_c is the center edges of text line 2 to $n-1$, and E_r is the right edges of text line 1 to $n-1$. We first estimate the median of E_l, then compute the absolute deviation D_l of the elements of E_l from its median,

$$D_l = \{d_i|d_i = |e_{li} - \text{median}(E_l)|, 2 \leq i \leq n\}.$$

Similarly, we estimate the absolute deviation of the center edges and right edges: D_c and D_r. Then, we compute the probability of B being left, center, right, or both justified by observing the mean absolute deviation of the left, center and right edges,

$$P(V_{TXT}(B)|Alignment(B)) = P(\text{mean}(D_l), \text{mean}(D_c), \text{mean}(D_r)|Alignment(B)). \quad (13)$$

4 Experimental Results

Our testing data set has $1,125$ document pages [12]. All of them are machine printed, noise free data. Among them, 565 pages are real data from different business and law books. Another 560 pages are synthetic data generated using the method stated in Wang et. al. [12]. A hold-out cross validation experiment [2] was conducted on all the data with $N = 9$. Discrete lookup tables were used to represent the estimated joint and conditional probabilities used at each of the algorithm decision steps.

Suppose we are given two sets $\mathcal{G} = \{G_1, G_2, ..., G_M\}$ for ground-truthed foreground table related entities, e.g. cell entities, and $\mathcal{D} = \{D_1, D_2, ..., D_N\}$ for detected table related entities. The algorithm performance evaluation can be done by solving the correspondence problem between the two sets. Performance metrics developed in [6] can be directly computed in each rectangular layout structure set. The performance evaluation was done on cell level. The numbers and percentages of miss, false, correct, splitting, merging and spurious detections on real data set and on the whole data set are shown in Table 1. Compared with our early work [12], we improved the detection result rates from 90.32% to 95.67% and from 92.04% to 97.05% in the whole data set. For the real data set, we improved the detection result rates from 89.69% to 96.76% and from 93.12% to 93.86%.

		Total	Correct	Splitting	Merging	Mis-False	Spurious
Real Data Set	Ground Truth	679	657 (96.76%)	3 (0.44%)	15 (2.21%)	4 (0.59%)	0 (0.00%)
	Detected	700	657 (93.86%)	6 (0.86%)	7 (1.00%)	30 (4.29%)	0 (0.00%)
Whole Data Set	Ground Truth	10934	10461 (95.67%)	132 (1.21%)	45 (0.41%)	296 (2.71%)	0 (0.00%)
	Detected	10779	10461 (97.05%)	264 (2.45%)	18 (0.17%)	36 (0.33%)	0 (0.00%)

Table 1. Cell level performance of the table detection algorithm on real data set and whole data set.

Figure 4 shows a few table detection examples. Figure 4(a), (b) are the correct detection results of Figure 1(a), (b), respectively. In Figure 4(a), our algorithm grows the original table and include its lower neighbor and construct a new table entity. In Figure 4(b), our algorithm eliminates the originally detected table and merge it with its lower neighbor and construct a new text block entity. Figure 4(c), (d) are our algorithms failed examples. Figure 4(c) shows a false alarm example. Some text in a figure are detected as a table entity. Figure 4(d) shows an error example where our table decomposition algorithm failed.

5 Conclusion and Future Work

In this paper, we formulated table detection problem in the whole page segmentation framework. We tried to improve table detection result by optimizing the whole page segmentation probability, including table entities and text block entities. We used iterative updating method to incrementally optimize the probability. We implemented our algorithm and tested on a data set which includes $1,125$ document pages with $10,934$ table cell entities. Among them, 565 pages are real data from different business and law books. Another 560 pages are synthetic data generated using the method stated in Wang et. al. [12]. Our experimental results demonstrated the correctness of our proposed algorithm.

As shown in Figure 4(d), our current table decomposition algorithm needs to be refined. We will develop a statistical based table decomposition algorithm. Our formulation has the potential to be useful to other more general page segmentation problems, for example, to segment text, figure and images, etc. We will study it in the future.

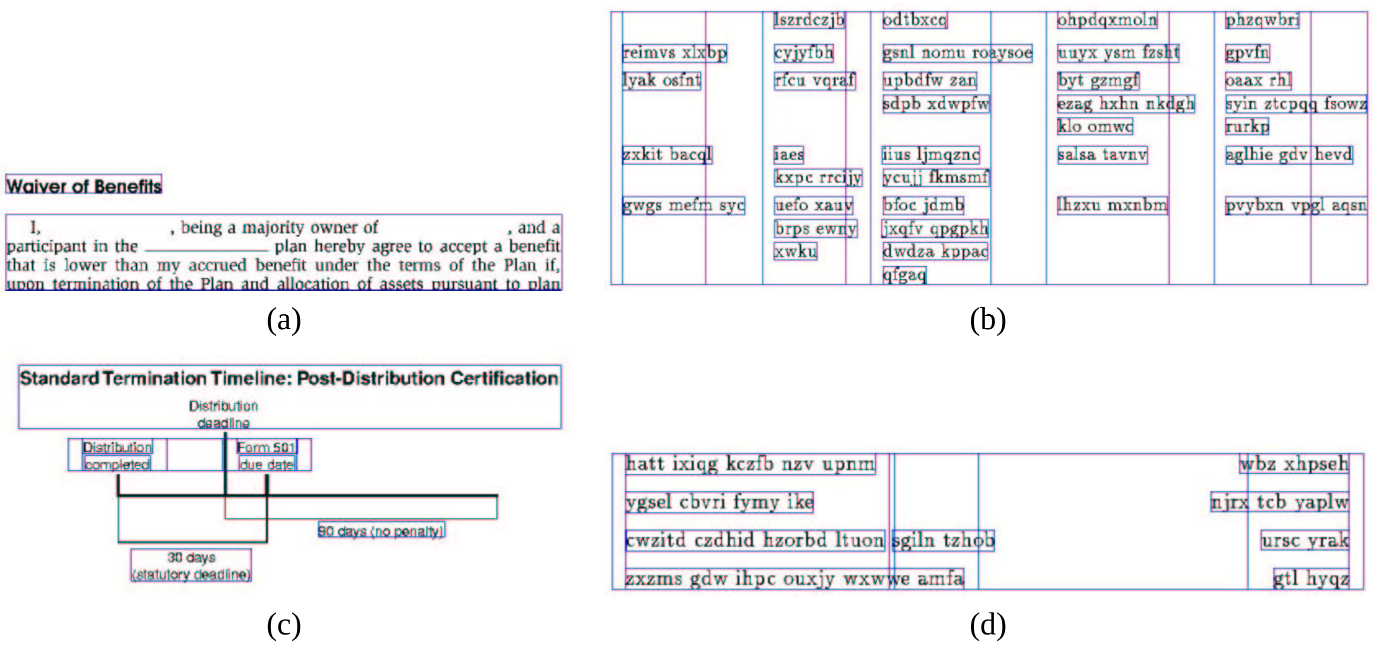

Waiver of Benefits
I, , being a majority owner of , and a participant in the plan hereby agree to accept a benefit that is lower than my accrued benefit under the terms of the Plan if, upon termination of the Plan and allocation of assets pursuant to plan
(a)
(b)
Standard Termination Timeline: Post-Distribution Certification
Distribution deadline
Distribution completed
Form 501 due date
90 days (no penalty)
30 days (statutory deadline)
(c)
(d)

Figure 4. (a), (b) Correct table detection results; (c), (d) failed table detection results.

6 Table Detection Algorithm

Figure 5 shows the our algorithm diagram. Given a labelled page, first we will estimate its probability. For each table, we will consider several adjustments, which are to keep it as a table, to grow the table to include its upper/lower neighbor, to merge the table with its upper/lower neighbor and label it as text block. For each table, we compute the new probability under such adjustments. We select the adjustment which produces the biggest improvement upon the initial page segmentation probability. Repeat this process until no improvement can be made. The details of the algorithm are described as below.

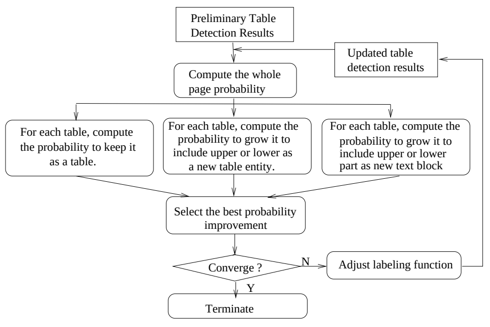

Preliminary Table Detection Results
Updated table detection results
Compute the whole page probability
For each table, compute the probability to keep it as a table.
For each table, compute the probability to grow it to include upper or lower as a new table entity.
For each table, compute the probability to grow it to include upper or lower part as new text block
Select the best probability improvement
Converge ?
N
Adjust labeling function
Y
Terminate

Figure 5. Overview of the table detection algorithm

Algorithm 6.1 *Table Optimization*

1. The input data to the algorithm are our previous table detection [12] and text block segmentation results.

2. For each table, i, $i = 1, ..., N$, where N is the number of tables in the given page. Compute the different probabilities under different adjustments. Compared with the original labeling results, only one adjustment is taken each time.

   - Keep the table. Compute the probability $P_{(i,1)}$ following Equation 12;
   - Merge table i with its upper text neighbor and label it as a new table. Compute the new probability $P_{(i,2)}$ following Equation 12;
   - Merge table i with its upper text neighbor and label it as a new text block. Compute the new probability $P_{(i,3)}$ following Equation 12;
   - Merge table i with its lower text neighbor and label it as a new table. Compute the new probability $P_{(i,4)}$ following Equation 12;
   - Merge table i with its lower text neighbor and label it as a new text block. Compute the new probability $P_{(i,5)}$ following Equation 12.

3. Compute $P_{max} = MAX(P_{(i,j)}), i = 1, ..., N, j = 1, ..., 5$ and get its appropriate adjustment *action*.

4. If *action* is to keep the table, then, return the labeling result and terminate the algorithm.

   Else take the adjustment *action* and go back to 2.

References

[1] E. Green and M. Krishnamoorthy. Model-based analysis of printed tables. *Proceedings of the 3rd ICDAR*, pages 214–217, August 1995.

[2] R. Haralick and L. Shapiro. *Computer and Robot Vision*, volume 1. Addison Wesley, 1997.

[3] J. Hu, R. Kashi, D. Lopresti, and G. Wilfong. Medium-independent table detection. *SPIE Document Recognition and Retrieval VII*, pages 291–302, January 2000.

[4] T. G. Kieninger. Table structure recognition based on robust block segmentation. *Document Recognition V.*, pages 22–32, January 1998.

[5] G. E. Kopec and P. A. Chou. Document image decoding using markov source models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16:602–617, June 1994.

[6] J. Liang. *Document Structure Analysis and Performance Evaluation*. Ph.D thesis, Univ. of Washington, Seattle, WA, 1999.

[7] J. Liang, I. T. Phillips, and R. M. Haralick. Consistent partition and labeling of text blocks. *Journal of Pattern Analysis and Applications*, 3:196–208, 2000.

[8] H. T. Ng, C. Y. Lim, and J. L. Koo. Learning to recognition tables in free text. *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 443–450, 1999.

[9] J. H. Shamilian, H. S. Baird, and T. L. Wood. A retargetable table reader. *Proceedings of the 4th ICDAR*, pages 158–163, August 1997.

[10] T. A. Tokuyasu and P. A. Chou. An iterative approach to document image analysis. *DLIA99 workshop*, September 1999.

[11] Y. Wang, R. Haralick, and I. T. Phillips. Improvement of zone content classification by using background analysis. *Fourth IAPR International Workshop on Document Analysis Systems. (DAS2000)*, December 2000.

[12] Y. Wang, I. T. Phillips, and R. Haralick. Automatic table ground truth generation and a background-analysis-based table structure extraction method. *Sixth International Conference on Document Analysis and Recognition(ICDAR01)*, pages 528–532, September 2001.