2013 12th International Conference on Document Analysis and Recognition

Learning to Detect Tables in Scanned Document Images using Line Information

T Kasar*, P Barlas*, S Adam*, C Chatelain† and T Paquet*
*Laboratoire LITIS - EA 4108, Universite de Rouen, FRANCE 76800
Email: thotreingam.kasar@inv.univ-rouen.fr, philippine.barlas@insa-rouen.fr,
{Sebastien.Adam,Thierry.Paquet}@litislab.eu
†Laboratoire LITIS - EA 4108, INSA Rouen, FRANCE 76800
Email: clement.chatelain@insa-rouen.fr

***Abstract*—This paper presents a method to detect table regions in document images by identifying the column and row line-separators and their properties. The method employs a run-length approach to identify the horizontal and vertical lines present in the input image. From each group of intersecting horizontal and vertical lines, a set of 26 low-level features are extracted and an SVM classifier is used to test if it belongs to a table or not. The performance of the method is evaluated on a heterogeneous corpus of French, English and Arabic documents that contain various types of table structures and compared with that of the Tesseract OCR system.**

***Keywords*-Table detection; line detection;**

I. Introduction

Tables are compact and efficient for summarizing relational information present in diverse document classes such as newspaper articles, scientific papers, forms, invoice, product descriptions or financial statements. While it is relatively easy for humans to spot a table, a precise definition of a table still remains elusive. Due to the inumerable possible types of table layouts, it is difficult to model a table and automatic understanding of tables from generic documents still remains a daunting task.

The process of automatic understanding of tables involves the following two modules i) table region detector that identifies regions in the document that correspond to tables and ii) table structure recognizer that extracts relational information from the identified table region to derive the logical structure of the table; direct application of optical character recognition (OCR) will simply fail since the fields of a table are inter-related and individually carry a little sense. The focus of our work in this paper is on the problem of table detection. Just like the need for preprocessing steps like skew correction or text-graphics separation in any OCR system, localizing table regions is also an indispensable step to ensure higher success rates for the subsequent processing stages. Most research on table recognition assume that the table region is already known, and focus only on extracting its logical structure. Robust segmentation of table regions is necessary to ensure a more reliable table recognition.

II. Review of related work

Tables can occur in different types of media [1], [2] such as ASCII, HTML or PDF. However, this work is focussed only on tables that occur in printed document images. One of the earliest works on identifying tabular regions in document images is the method proposed by Watanabe et al. [3]. The method identifies individual item blocks enclosed by vertical and horizontal line segments. Firstly, line segments are detected and the corner points are subsequently determined. The connective relationships among the extracted corner points and hence the individual item blocks are interpreted using global and local tree structures.

Laurentini and Viada [4] proposed a method to detect tables where text and lines are horizontal or vertical. The arrangement of detected lines is compared with that of the text blocks in the same area. Further, using the horizontal and vertical projection profiles, the algorithm attempts to add missing horizonal and vertical lines in order to fully understand the table structure. Green and Krishnamoorthy [5] proposed a model-based top-down approach for table analysis by a hierarchical characterization of the physical cells. Horizontal lines, vertical lines, horizontal space and vertical space are used as features to extract the table region. Cesarini et al. [6] present a system for locating tables using a recursive analysis of the modified X-Y tree to identify regions enclosed by horizontal (vertical) lines. The search is refined by looking for further parallel lines that can be found in deeper levels of the tree. Sub-tables belonging to one table are merged while tables smaller than a given threshold were discarded. It requires that at least two parallel lines are present. Gatos et al. [7] detect horizontal and vertical rulings and progressively identify all possible types of line intersection. Table reconstruction is then achieved by drawing the corresponding horizontal and vertical lines that connect all line intersection pairs.

Hu et al. [9] presented a system for table detection from scanned images and ASCII documents. It uses an optimization technique to identify textlines belonging to a table. However, it assumes a single-column input page that can be easily segmented into individual textlines. In [8], the authors propose a method that can handle multi-column pages using the layout analysis module of Tesseract OCR. Table regions are determined using certain heuristic rules based on analysis of the column layout of the page and the column partitions. However, it requires the presence of large

1520-5363/13 $26.00 © 2013 IEEE
DOI 10.1109/ICDAR.2013.240

text regions (paragraphs) so that the column layouts can be reliably estimated.

Methods such as the ones proposed in [11], [12] do not rely on the presence of lines but use only text information. In [11], tables are assumed to have distinct columns so that the gaps between the fields are substantially larger than the inter-word gaps in normal text lines. It works only for Manhattan layout and may fail for complex documents. All lines are removed as a pre-processing step. This can result in inaccurate detections for partially-filled tables. In [10], the authors proposed a method to detect tables in handwritten document images. Text regions are first identified using an SVM classifier and then table regions are determined based on a correlation score between adjacent text lines. Like other text-based approaches, the detected regions even for correct detections can still have large discrepancies when compared with the ground-truth.

Most of the existing approaches focus on table recognition to extract the logical structure of the tables and hence use simple heuristic rules to detect tables. Such an approach may fail to detect complex table structures. Existing methods also assume specific page layouts and hence multi-column documents or the presence of handwritten elements require special treatment. We propose a versatile method that overcomes these limitations by adopting a learning-based approach to classify tables using line information.

III. System description

Our method employs a classifier to detect tables that is learned from an annotated database. By doing so, the method is able to handle the intrinsic variability of the table structures without resorting to user-defined heuristic rules. Moreover, since the method does not rely on the text information, it is also insensitive to the script or the layout of the page and can handle multi-column documents. Another advantage of the method is that it does not need any handwritten or printed text analysis, which are usually difficult, error-prone and time consuming. In our approach, we seek to identify horizontal and vertical lines present in the image and use a learned classifer to locate tables based on the properties of the detected lines. A schematic block-diagram of the proposed method is shown in Figure 1.

A. Extraction of horizontal and vertical lines

We employ a run-length approach to extract lines along the horizontal and vertical directions. In this work, it is assumed that the table is printed on a white background. So, in order to enhance dark and thin line-like structures, the input image I is first smoothed with a Gaussian filter and the grayscale morphological black-hat operation is then performed.

$$I_\sigma = I * G_\sigma \tag{1}$$

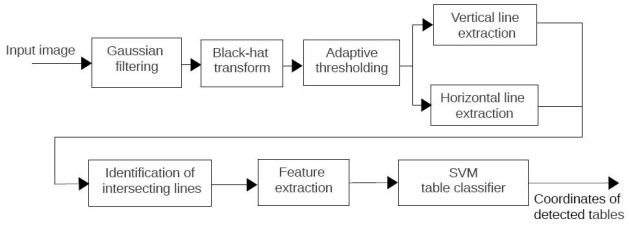


Figure 1. Schematic block diagram of the proposed method.

$$I_p = (I_\sigma \bullet S_N) - I_\sigma \tag{2}$$

Where σ represents the variance of the Gaussian filter, S_N is a square structuring element of size N, $*$ and $\bullet$ denote the 2-D convolution and the morphological closing operation respectively. The Gaussian smoothing operation helps to maintain the continuity of narrow gaps between line segments whereas the effect of black-hat operation is to highlight 'small' dark structures while suppressing 'wide' dark structures at the same time. The variance σ of the Gaussian function controls the amount of smoothing. The size N of the structuring element decides the maximum width of the line that can be detected by the system and is empirically set to 7 in this work.

The pre-processed image I_p is then thresholded adaptively with k_1 times its maximum intensity as the threshold value. The method does not require accurate binarization since it relies only on the line information and not on the textual components. The threshold k_1 is fixed to a low value of 0.05 so that even 'weak' lines show up after thresholding. The resulting binary image I_{bw} is subjected to a run-length count along the rows and columns to obtain horizontal and vertical lines. If the count of 'ON' pixels in a particular direction starting at a pixel location exceeds a threshold value l, the segment is accepted as a line. All pixels with run-lengths less than the specified threshold value are ignored. The threshold l decides the shortest line that can be detected by the system and is adaptively set to 1/20 times the width of the input image. Since tables normally occupy a significant area of the image, the method is insensitive to the choice of this threshold.

It may be mentioned here that the document is assumed to have no skew, and hence a skew detection and correction step may be invoked, if necessary. Combining the output of the line segments obtained from the two directions, we get a composite image I_L that contains all the detected horizontal and vertical lines.

B. Extraction of table features

We perform a connected component (CC) analysis on the line image I_L and proceed to validate those CCs that are comprised of at least 3 intersecting horizontal (or vertical)

lines as tables or not. Let $H_i, i = 1, 2, n$ and $V_j, j = 1, 2, m$ denote the horizontal and vertical lines that constitute the CC respectively. The indices i of the horizontal lines are sorted in top-to-bottom order while the indices j sorted in left-to-right order. Let L_H^i, H_{start}^i and H_{end}^i represent the length, starting and ending positions of the line H_i respectively. The inter-line spacing between H_{i+1} and H_i is given by the L_∞-norm between the starting (or ending) points of the two lines; $H_{spacing}^{i,i+1} = |H_{start}^{i+1} - H_{start}^i|_\infty$. Similarly, $L_V^j, V_{start}^j, V_{end}^j$ and $V_{spacing}^{j,j+1}$ denote the same for the vertical line counterparts.

In order to locate tables that are enclosed by lines, the following features are computed from each CC (group of intersecting horizontal and vertical lines):

$$f_1 = \frac{L_H^1}{Max(\{L_H\})} \quad (3)$$

$$f_2 = \frac{L_H^n}{Max(\{L_H\})} \quad (4)$$

$$f_3 = \frac{|H_{start}^n - H_{start}^1|_\infty}{Max(\{L_V\})} \quad (5)$$

$$f_4 = \frac{L_V^1}{Max(\{L_V\})} \quad (6)$$

$$f_5 = \frac{L_V^m}{Max(\{L_V\})} \quad (7)$$

$$f_6 = \frac{|V_{start}^m - V_{start}^1|_\infty}{Max(\{L_H\})} \quad (8)$$

The above features try to ensure that the first and the last horizontal and vertical lines completely enclose the table region. These features also inhibit non-table CCs with spurious 'long' lines in the mid-region.

The lines that constitute a table normally have some degree of regularity in their relative position, length and spacing between adjacent ones. Tables also occupy a significant portion of the image area in general. These characteristics are captured by the following features:

$$f_7 = \frac{Median(\{L_H\})}{Max(\{L_H\})} \quad (9)$$

$$f_8 = \frac{Std(\{L_H\})}{Mean(\{L_H\})} \quad (10)$$

$$f_9 = \frac{Std(\{H_{spacing}\})}{Mean(\{H_{spacing}\})} \quad (11)$$

$$f_{10} = \frac{Std(\{H_{start}\})}{Mean(\{H_{start}\})} \quad (12)$$

$$f_{11} = \frac{Std(\{H_{end}\})}{Mean(\{H_{end}\})} \quad (13)$$

$$f_{12} = \frac{Median(\{L_V\})}{Max(\{L_V\})} \quad (14)$$

$$f_{13} = \frac{Std(\{L_V\})}{Mean(\{L_V\})} \quad (15)$$

$$f_{14} = \frac{Std(\{V_{spacing}\})}{Mean(\{V_{spacing}\})} \quad (16)$$

$$f_{15} = \frac{Std(\{V_{start}\})}{Mean(\{V_{start}\})} \quad (17)$$

$$f_{16} = \frac{Std(\{V_{end}\})}{Mean(\{V_{end}\})} \quad (18)$$

$$f_{17} = \frac{Height(CC)}{Height(InputImage)} \quad (19)$$

$$f_{18} = \frac{Width(CC)}{Width(InputImage)} \quad (20)$$

In addition, since the constituent lines of a table intersect each other, a CC belonging to a table will exhibit a high negative value of Euler number due to the presence of a large number of cells. Tables are normally enclosed by lines on both horizontal and vertical sides. Unlike tables, graphic objects and line-drawings tend to have 'open' lines that do not intersect other lines at one of its end. Due to the presence of 'open' lines, such non-table objects tend to have a much higher number of convex deficiency regions. These characteristics are captured by the following features:

$$f_{19} = \#OpenHor.Lines \quad (21)$$

$$f_{20} = \#OpenVert.Lines \quad (22)$$

$$f_{21} = EulerNumber(CC) \quad (23)$$

$$f_{22} = \#IntersectionPoints \quad (24)$$

$$f_{23} = \frac{\#(CCBordersPixels == ON)}{Perimeter(BoundingBox(CC))} \quad (25)$$

$$f_{24} = \#ConvexDeficiencyRegions \quad (26)$$

$$f_{25} = \frac{Area(ConvexDeficiencyRegions)}{Area(Imfill(CC, holes))} \quad (27)$$

$$f_{26} = \frac{Perimeter(ConvexHull(CC))}{Perimeter(CC)} \quad (28)$$

C. Table classification with an SVM classifier

We use the LibSVM toolbox [13] to implement an SVM classifier using the above features. The classifier is trained using an RBF kernel on a annotated dataset containing table and non-table examples. The optimal parameters of the SVM classifier are obtained using a grid-search with 5-fold cross-validation process.

For a given test image, lines are detected as described in Section III-A and each group of intersecting horizontal and vertical lines are validated using the trained classifier. Isolated lines and CCs that contain less than 3 intersecting horizontal or vertical lines are not considered for table classification.

IV. Experimental results

We use the MAURDOR campaign dataset [14] 'train0' for training and 'dev1' for testing. Each of the datasets comprises of a heterogeneous corpus of 1000 scanned documents

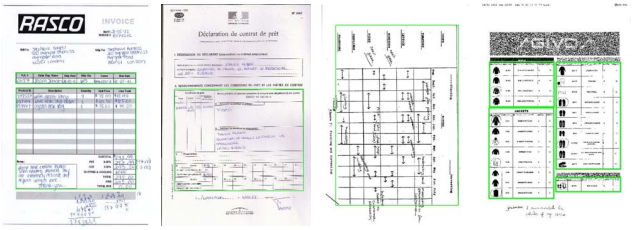


Figure 2. Representative sample images of the test data shown along with the detected tables.

and fax images. The dataset contains text in 3 different scripts namely, French, Arabic and English with various types of table structures. These documents also contain both printed and handwritten text. Some representative examples of the test data are shown along with the detected table regions in Figure 2.

The ground-truth is represented in an XML file where the table regions are given by the coordinates of its circumscribing rectangle. To evaluate the performance of our method, we use the evaluation metrics employed in [8], [10]. For the sake of completeness, these metrics are described again. Let D_i represent the bounding box of the i^{th} detected table and G_j represent the bounding box of j^{th} ground-truth table. Then, the amount of area overlap between the two is given by:

$$A(i,j) = \frac{2|D_i \cap G_j|}{|D_i| + |G_j|} \tag{29}$$

where $|D_i \cap G_j|$ represents the area of intersection of the two rectangles, $|D_i|$ and $|G_j|$ denote the individual area of the detected rectangle and the ground-truth respectively. Any resulting indeterminate 0/0 form is deemed to be 0. Based on the amount of area overlap, the following metrics are defined:

Correct: The number of detected tables that have a one-to-one correspondence with ground-truth tables with $A \geq 0.9$.

Partial: The number of detected tables that have a one-to-one correspondence with ground-truth tables but $(0.1 < A < 0.9)$.

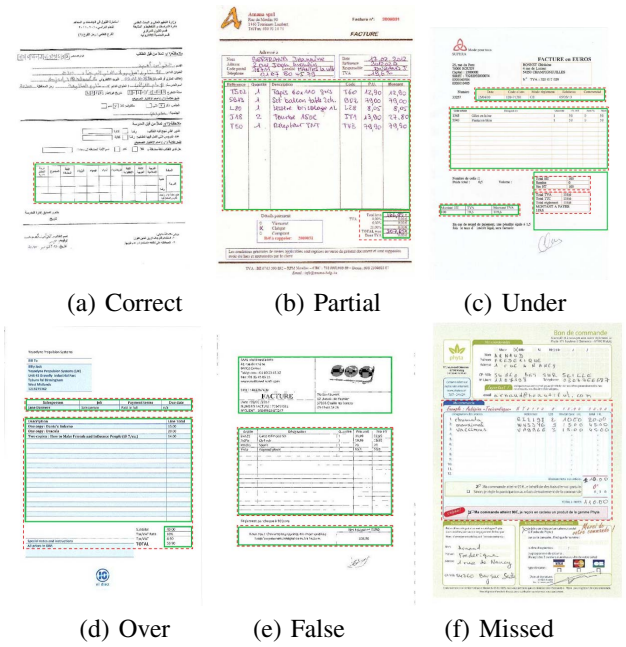
Under: The number of detected tables that have a major overlap $(0.1 < A < 0.9)$ with more than one ground-truth tables.

Over: The number of detected tables that have a major overlap $(0.1 < A < 0.9)$ with one ground-truth table, but the corresponding ground-truth table has major overlaps with other detected tables as well.

False: The number of detected tables that do not overlap with any of the ground-truth tables $(A \leq 0.1)$.

Missed: The number of ground-truth tables that do not overlap with any of the detected tables $(A \leq 0.1)$.

An example instance of each type of detections are shown in Figure 3. Since our method relies only on the line information, we conduct the following two experiments:



(a) Correct (b) Partial (c) Under

(d) Over (e) False (f) Missed

Figure 3. Example instances of different types of performance measures. The red dashed boxes represent the ground truth while the table regions detected by our method are represented by solid green rectangles.

i) With a selected subset of 200 images: These images contain 308 tables that are enclosed by straight lines. Our method yields 288 detections, out of which 250 are Correct; 10 Partial; 5 Under; 2 Over; 40 Missed and 21 False.

ii) With the complete dataset of 1000 documents: There are actually 1077 images since some documents have multiple pages. Out of these, 233 images contain at least one table, and a total of 347 tables in all. It may be noted that this includes tables that do not have proper border lines. Our method yields 346 detections, out of which 250 are Correct; 12 Partial; 5 Under; 2 Over; 77 Missed and 77 False.

The difference in the number of Missed detections is due to tables without border lines. We see that the method is quite robust, with an increase of just 56 instances of False detections considering the fact that the dataset contains numerous other entities such as line-drawings, pictures, hand-drawn sketches and form-fields.

We also employ precision and recall measures that are widely used in the information retrieval community. The area precision is defined as the ratio of the area of the detected regions that actually belong to table regions to the total area of the detected regions while the area recall gives the percentage of the ground-truth table regions that are detected as tables by the method. These metrics are computed as follows:

$$AreaPrecision = \frac{|D \cap G|}{|D|} \tag{30}$$

Table I
COMPARATIVE RESULTS OF THE PROPOSED METHOD AND TESSERACT TABLE DETECTOR

Metric	Tesseract		Proposed method	
	Complete test set	Subset of 200 images	Complete test set	Subset of 200 images
Correct	32	30	250	250
Partial	106	89	12	10
Under	22	15	5	5
Over	30	26	2	2
Missed	135	126	77	40
False	326	28	77	21
Precision	-	58.9%	-	83.9%
Recall	-	40.1%	-	84.1%

$$AreaRecall = \frac{|D \cap G|}{|G|} \quad (31)$$

Instances of indeterminate 0/0 form are taken to be 0. However, the notion of precision and recall requires that there is at least one table present in the groundtruth i.e. $|G| \neq 0$. Therefore, they are computed only for the selected subset of images. The average precision and recall values obtained are 83.9% and 84.1% respectively.

The performance of our method is also compared with that of the table detector of Tesseract OCR system on the same data and the results obtained are summarized in Table I. It is observed that the Tesseract table detector rely on the text information, which results in large localization errors; clearly evident from the large number of partial detections. Also, a large number of tables are missed by Tesseract since it is difficult to localize all the types of tables present in the dataset using the text information alone. This is also reflected in the relatively lower values of the average precision and recall. On the other hand, our method yields accurate table detections with significantly fewer instances of missed detections and false positives.

V. DISCUSSION AND FUTURE WORK

We have presented a new method for table detection in scanned document images using the properties of line separators. The method yields promising results on a heterogeneous collection of documents. It uses a learned classifier to classify table regions without resorting to heuristic rules. Unlike other image-based techniques, our method relies only on the presence of lines and do not require any text analysis. This makes it insensitive to the layout, presence of multiple scripts and handwritten elements. It does not require a specialized binarization process and all the lines can be detected reliably thanks to the action of morphological black-hat operation. Since we employ run-length to obtain the lines, the method is also not affected by characters and other objects that touch the lines. Identification of the row and column line separators during the table detection stage implicitly gives the information about each cell of the table which can be utilized in subsequent processing steps of understanding its logical structure.

While the method performs well for tables completely enclosed by lines, there are, in practice, other types of table layouts that may contain only parallel lines that separate the table header or with no lines at all. Our future work is to extend the method to handle tables without border lines too using additional cues from the image such as the arrangement of text and white spaces.

REFERENCES

[1] R. Zanibbi, D. Blostein and J. R. Cordy, *A survey of table recognition - Models, observations, transformations, and inferences*, Intl. Jl. Doc. Anal. Recog., 7, 1-16, 2003.

[2] D. Embley, M. Hurst, D. Lopresti and G. Nagy, *Table processing paradigms: a research survey*, Intl. Jl. Doc. Anal. and Recog., 8(2), pp. 66-86, 2006.

[3] T. Watanabe, H. Naruse, Q. Luo and N. Sugie, *Structure analysis of table-form documents on the basis of the recognition of vertical and horizontal line segments*, Proc. Intl. Conf. Doc. Anal. Recog., pp. 638-646, 1991.

[4] A. Laurentini and P. Viada, *Identifying and understanding tabular material in compound documents*, 2, Proc. Intl. Conf. Patt. Recog., 1992.

[5] E. A. Green and M. S. Krishnamoorthy, *Model-Based Analysis of Printed Tables*, Proc. Intl. Conf. Doc. Anal. and Recog., pp. 214-217, 1995.

[6] S. Cesarini, S. Marinai, L. Sardi and G. Sorda, *Trainable table location in document images*, Proc. Intl. Conf. Patt. Recog., pp. 236-240, 2002.

[7] B. Gatos, D. Danatsas, I. Pratikakis and S. J. Perantonis, *Automatic table detection in document images*, Proc.Intl. Conf. Advances Patt. Recog., pp. 609-618, 2005.

[8] F. Shafait and R. Smith, *Table Detection in Heterogeneous Documents*, Proc. Intl. Workshop. Doc. Anal. Sys., pp. 65-72, 2010.

[9] J. Hu, R. Kashi, D. Lopresti, and G. Wilfong, *Medium-independent table detection*, Proc. SPIE Doc. Recog. and Retr. VII, pp. 291-302, 2000.

[10] J. Chen and D. Lopresti, *Table detection in noisy offline handwritten documents*, Proc. Intl. Conf. Doc. Anal. and Recog., pp. 399-403, 2011.

[11] S. Mandal, S. P. Chowdhury, A. K. Das, B. Chanda, *A simple and effective table detection system from document images*, Intl. Jl. Doc. Anal. and Recog., 8(2), pp. 172-182, 2006.

[12] T. G. Kieninger, *Table structure recognition based on robust block segmentation*, Proc. Doc. Recog. V, SPIE, 3305, pp. 22-32, 1998.

[13] C. C. Chang and C. Lin, *LIBSVM: a library for support vector machines*, http://www.csie.ntu.edu.tw/cjlin/libsvm.

[14] MAURDOR campaign dataset, http://www.maurdor-campaign.org/