On Segmentation of Documents in Complex Scripts

K. S. Sesh Kumar, Sukesh Kumar and C. V. Jawahar
Centre for Visual Information Technology
International Institute of Information Technology, Hyderabad, India - 500032
jawahar@iiit.ac.in

Abstract

Document image segmentation algorithms primarily aim at separating text and graphics in presence of complex layouts. However, for many non-Latin scripts, segmentation becomes a challenge due to the characteristics of the script. In this paper, we empirically demonstrate that successful algorithms for Latin scripts may not be very effective for Indic and complex scripts. We explain this based on the differences in the spatial distribution of symbols in the scripts. We argue that the visual information used for segmentation needs to be enhanced with other information like script models for accurate results.

1. Introduction

Segmentation is an important intermediate step in any document image analysis algorithm. Segmentation algorithms extract homogeneous regions of text, graphics etc. from document images. Subdividing the text into words is of paramount importance in recognition systems. Recent segmentation algorithms aim at performing the geometric layout analysis of the document images even when the layouts are complex [1, 3, 12]. Since it is perceived that layouts can be analyzed independent of the script, most of the reported algorithms are demonstrated successfully on segmentation of documents in Roman scripts [3, 12].

With increased interests in document images of digital libraries, segmentation of documents in many non-Latin scripts have become an immediate requirement. Robust segmentation of images into words is also identified as a blockade in the development of OCRs for these languages. In this paper, we argue that the script, an important characteristic of the document image, which needs to be taken into consideration while designing the segmentation algorithm.

A class of algorithms for segmenting the image into text and graphics employ texture features [9]. They provide an immediate framework for adding the script specific information into the segmentation algorithm because texture is one of the critical clues in demarcation of scripts [15]. We do not attempt this task. Below the block-level, texture clues are not immediately applicable, structural and other shape based measures are more effective. Our interest is limited to a narrow spectrum of the segmentation tasks. We focus on seemingly simple looking, but critically important, task of segmenting textual blocks into constituent words. We empirically evaluate six popular segmentation algorithms on documents of five scripts. We analyze the failure reasons and provide the direction to overcome this.

Role of Script: The spatial nature of a text block can be estimated by analyzing its Distance-Angle plots. These plots give an overview of the distances of neighboring components and the angles at which they are available. Figure-1 shows that in English documents, the components either lie at an angle of ± 90 degrees or 0 degrees with the horizontal. However in scripts such as Telugu, no such structural nature could be observed. Hence segmentation algorithms designed with such structural assumptions of text blocks fail to give good results on other text blocks.

The shape of connected components also provides critical information for the segmentation of text blocks. In roman scripts, components do not vary much in shape or size. This nature of the script helps in segmentation. However this is not possible in Telugu because the connected component sizes vary highly as shown in Figure-1.

Roman scripts and a few scripts like Hindi, Bangla use a four ruled standard (see Figure 2). Lack of such standards in scripts like Telugu and Urdu introduces complexity into the segmentation of text block. In these scripts the component boundaries also occasionally overlap with the boundaries of the neighboring lines as shown in Figure 3. Hence the bounding box based segmentation algorithms [4, 14] give acceptable segmentation results on Roman and Devanagari scripts but fail to give convincing results on scripts like Telugu, Urdu etc. The complexity of documents is due to the spatial distribution of connected components. Thus script also provides valuable information about the document which can be used for segmentation of the document.

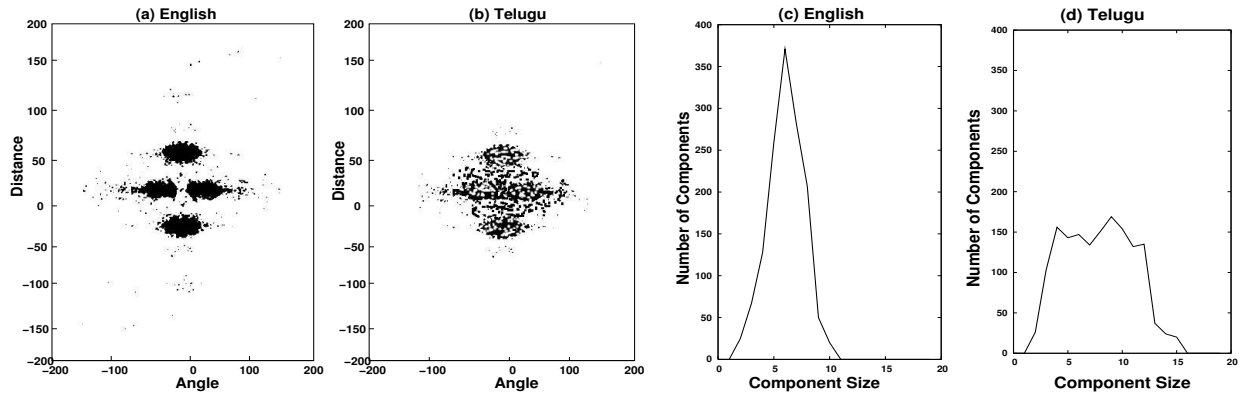

Figure 1. Distance-Angle plot nearest neighbor components (a) and (b): Note the clusters of nearest neighboring components at angles ± 90 and 0 degrees with the horizontal in English. Component Size Graphs in (c) and (d): Note that the component sizes vary widely in Telugu compared to English.

2. Segmentation Algorithms

There are a large number of segmentation algorithms available in literature. Some of the standard page segmentation algorithms are selected and their performance on documents with complex layouts is compared and analyzed [17]. We use these algorithms to analyze their performance on documents with complex scripts.

A1-Recursive XY CUT: This algorithm [13] is a tree based segmentation algorithm. A document is recursively split horizontally and vertically until a final criteria where a split is impossible is met. The document forms the root of the tree while each split expands the tree. The nodes form the complete segmentation of the document.

A2-Whitespace Analysis: The algorithm [4] attempts to form the maximal whitespace rectangles in the document. These whitespace blocks are merged to form the non text regions within the document. Various methods of sorting the white space rectangles are used to find the white rectangle covers. The covers are accepted based on the weights that are assigned to acceptable rectangles.

A3-Constrained text-line Detection: The constrained text line detection algorithms [2] are another set of top down algorithms, which find the gutters of white spaces in the document. The white space rectangles are allowed to overlap within a particular threshold. They are sorted in order of decreasing quality, which are further used to find the text lines within the document image.

A4-Docstrum: It is a bottom up segmentation algorithm [14]. The components are merged together based on a nearest neighbor based approach. This algorithm heavily depends on the thresholds that are set for a document. It clusters all the nearest neighbor components into a particular region which are further clustered into the regions of higher order (text blocks).

A5-Voronoi-diagram based algorithm: The voronoi based segmentation algorithm [8] extracts points on the boundaries of the connected components. The voronoi cells are drawn surrounding the points. Superfluous edges formed by these voronoi cells are removed using a criteria functions which involves thresholds like distance and ratio of areas of adjacent cells to form the components.

A6-Smearing: The run length smearing algorithm [19] is a bottom up approach. Bitmaps are formed by grouping the pixels horizontally and vertically using thresholds of acceptable distance between the components in each direction. The document is segmented by performing logical operations on the bitmaps. The performance of the algorithm highly depends on the thresholds used.

Performance Metrics: Quantitative comparison of segmentation algorithms may be obtained by evaluating appropriate performance metrics. A large number of evaluation schema are proposed to measure the capability of the segmentation algorithm for a document image. They are popular for different purposes in document understanding. Kanai *et al.*[6] proposed an evaluation schema based on the number of edit operations (insertions, deletions and moves). This performance metric was used to evaluate the automatic zoning accuracy of four commercial OCRs. Vincent *et al.*[16] proposed a bitmap level region based metric. This evaluates both text based and nontext based regions in the document image but highly dependent on pixel noise. Liang *et al.*[10] proposed a region area based metric. The overlap area of a groundtruth zone and a segmentation zone is used to compute the performance metric. Mao *et al.* [11] proposed an empirical evaluation of the segmentation algorithms in the presence of groundtruth and improved the performance of the algorithm with iterations. It automatically trains the algorithm with free parameters over iterations and improves the performance of the segmentation algorithm.

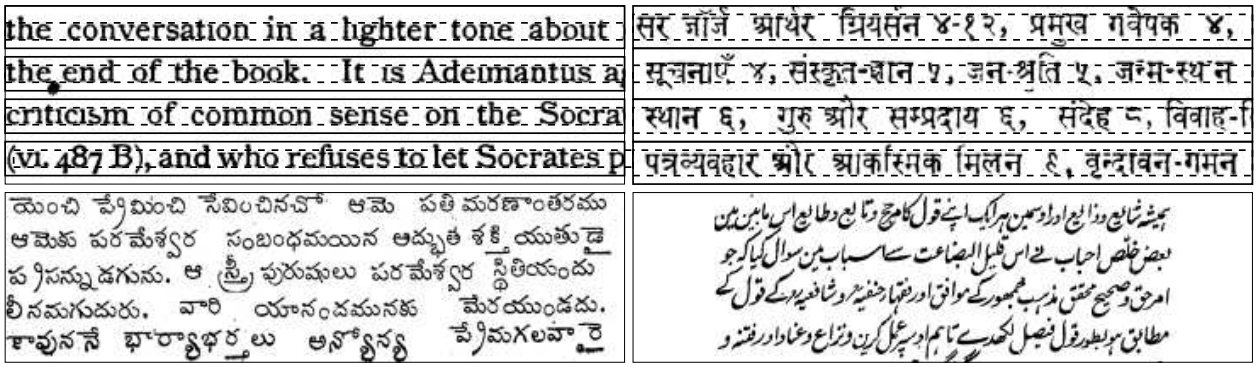

Figure 2. Observe the difference in writing styles and positioning of the symbols in different scripts

3. Empirical Evaluation

The above mentioned segmentation algorithms are mainly designed to use the visual information to perform segmentation We evaluate their performance on various datasets based on the performance metric proposed by Mao *et al.*[11]. They define the metric as $(n(L) - n(C_L \cup S_L \cup M_L \cup F_L))/n(L)$, where $n(.)$ gives the cardinality of the set, L is the set of ground truth text lines in the document, C_L is the set of missing ground truth text lines , S_L is the set of split ground truth text lines, M_L is the set of merged ground truth text lines, and F_L is the set of falsely detected noise zones. This measure gives the ratio of correctly detected lines to the total number of lines in the document. We use this measure for our experiments on English documents, which are a part of the CEDAR dataset. This dataset has large variations in terms of layouts. These documents also have a mixture of text blocks that belong to different fonts and font sizes within the document. The documents with Devanagari, Telugu, Tamil and Urdu are scanned from the books of respective scripts. These document images have a relatively simple layout with no variation in font sizes within the document. 50 documents are used for each of the scripts from different books. Across these documents there are variations in terms of fonts and font sizes.

The original datasets (**O**) are degraded synthetically using the document degradation models [7] to form the datasets: **D1**-Cuts, **D2**-Salt And Pepper, **D3**-Blobs and **D4**-Erosion. The performance of the standard segmentation algorithms were tested on these datasets. Every segmentation algorithm has a set of free parameters. The free parameters of the algorithms are set such that the segmentation gives the best results on the document image. Table- 1 shows the percentage of performance of various segmentation algorithms on the original Dataset-**O** and the degraded datasets-**D1, D2, D3, D4**.

It can be observed that the performance of the segmentation algorithm is reasonably good on documents with English scripts inspite of having complex layouts and high font variations within the document. The algorithms also performed robustly on the degraded datasets. The performance of the algorithms on documents with Devanagari script was good because of the simpler script nature and simple layouts. However, the algorithms failed to give acceptable performance on datasets with Telugu, Tamil and Urdu scripts inspite of very simple layouts and no font variations within the document. They could not perform well on their respective degraded datasets because of the script complexities within the document.

We further adapt the Docstrum, Recursive XY Cut and Voronoi algorithms to these scripts. The free parameters of the algorithms were selected such that the algorithm favors splits and the split lines and words were merged based on script specific post segmentation techniques. Improvements were observed in the performance of the algorithms on adaptation to the script specific structures. The Docstrum, Recursive XY Cut and Voronoi based algorithms on adapting to Devanagiri, Urdu, Telugu and Tamil datasets reported $(91.3, 60.0, 75.0, 86.7)$, $(91.0, 65.0, 75.6, 81.7)$ and $(92.0, 73.0, 76.1, 80.6)$ percent accurate segmentation on their respective datasets. There could be alternative approaches to provide similar results or further improvement [18]. More than the numerical values, these experiments demonstrate that segmentation of documents in complex scripts is still an unsolved problem. We discuss the conceptual reasons for the unsatisfactory segmentation results and possible solutions to the problem in the next section.

4. Discussions

Documents with simple scripts and complex layouts are majorly available document images. They have simple scripts like Roman scripts. In Roman scripts the connected components in a line are collinear. These connected components could be classified into ascenders, descenders and

Table 1. Performance of different segmentation algorithms on different datasets

Script		A1	A2	A3	A4	A5	A6
English	O	89.3	95.5	91.3	93.4	94.5	93.8
(CEDAR)	D1	85.2	93.0	90.2	93.0	93.1	92.7
	D2	86.2	92.0	91.0	92.0	93.5	93.6
	D3	85.3	93.8	89.3	91.3	91.6	94.3
	D4	85.8	94.1	90.1	92.5	92.3	92.1
Devanagari	O	86.4	91.5	93.5	90.1	91.5	92.5
	D1	83.5	90.1	91.0	88.6	89.5	87.4
	D2	84.2	91.0	92.1	87.5	88.6	89.4
	D3	81.5	89.8	90.5	89.5	91.0	91.4
	D4	84.0	88.5	89.5	90.0	90.5	90.0
Urdu	O	51.5	45.6	60.0	55.3	71.5	65.4
	D1	54.0	46.3	58.3	56.2	69.3	63.5
	D2	53.0	44.3	61.2	54.3	70.3	65.3
	D3	51.3	45.6	59.3	51.3	71.2	65.4
	D4	54.0	48.3	58.4	54.6	68.3	61.3
Telugu	O	71.5	73.2	75.3	69.6	75.6	74.3
	D1	70.3	72.2	72.8	68.5	74.9	73.6
	D2	69.6	72.3	74.6	69.3	73.6	71.3
	D3	71.3	72.9	71.5	69.4	74.2	74.2
	D4	70.9	71.7	73.6	67.5	75.0	70.0
Tamil	O	79.5	81.5	79.8	82.0	78.9	80.3
	D1	79.4	80.7	78.3	81.3	75.3	80.2
	D2	75.6	79.4	76.9	80.5	77.6	79.6
	D3	78.7	79.6	75.3	81.8	74.3	80.0
	D4	77.3	76.5	73.6	80.9	75.5	80.2

normal components based on the spatial position of their bounding box within the line. The complex layout and the font variations within a document form the major challenges in these document collections. The nearest neighbor based segmentation algorithms give good results on such collections because the neighboring components within a line can easily be estimated by plotting the Distance-Angle plots as shown in Figure 1. White space analysis based algorithms also work well on documents because it is easy to find maximal white rectangles due to the collinear nature of the connected components in a line.

The documents with complex scripts and simple layouts are the books that are scanned to be digitized. They do not have any complex layouts. In scripts like Telugu and Tamil, few components of a line drift away vertically from the line depending on the type of component. This effects the spatial distribution of components making the task of segmentation complex. The nearest neighbor based algorithms tend to fail on these documents due to the non-equidistant and overlapping nature of the connected components. The white space analysis based algorithms fail because it is not easy to find maximal white space rectangles as the dangling components lie between the lines. This results in poor maximal white space rectangles, which in turn result in poor segmentation.

Recursive XY Cut algorithm, a projection profile based approach depends on the white spaces between the lines. However there are components that lie between two lines due to the script nature. This results in a non-uniform projection profile which is not very easy to analyze. The threshold parameters that are adaptively calculated from these projection profiles do not give good results. Constrained line detection algorithm and the white space analysis based algorithm also fail on the documents because of the dangling components. They obstruct the white space resulting in poor maximal white rectangles. Hence the white rectangles thus obtained cannot result in poor maximal white cover or poor gutters.

Smearing, Docstrum and Voronoi based segmentation algorithms use nearest neighbor approaches of bounding boxes where the components belong to the line nearest to them. However, this does not hold for all scripts. A component which belongs to a particular line could be nearer to another line depending on the font nature of the script or the document nature. Docstrum also assumes an angle threshold for the "within line neighbors" which result in poor segmentation because in scripts such as Telugu, Tamil, Urdu etc. the components that belong to a particular line can lie at any angle. Each of these algorithms can be adapted to a particular document with a complex script and attain acceptable results. However, it is very likely to fail on a document which vary in font, size or script. It is because the algorithm does not have any form of contextual information to perform segmentation.

The possible solution to segment the documents with complex script is to provide more information because the segmentation algorithms designed based on visual information only are not very successful in segmenting these documents. More contextual information such as script specific information is needed to perform the accurate segmentation of the document. The contextual information can be expressed in the form of shape and the spatial distribution of connected components, which could be modeled. The models also could encapsulate the information which is not only script specific but also document specific. These models can form an important knowledge base for the task of segmentation. Hence, algorithms need to be designed to use contextual information for segmentation of documents. Constrained text line detection algorithm [18] is improved to segment Urdu documents using the information provided in the form of Urdu script nature. Script specific properties are learnt from a collection of documents of a particular script in the form of spatial language models to segment the other documents of the script [5].

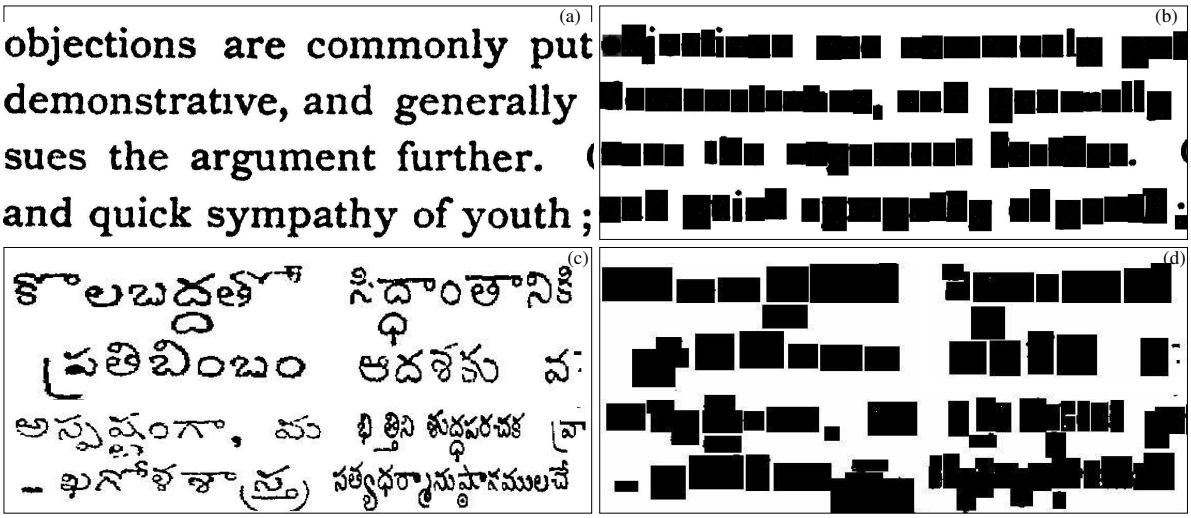

Figure 3. Note that the bounding boxes alone cannot be used to segment Telugu documents

5. Conclusion

We argued that popular segmentation algorithms are designed for documents with simple scripts. These algorithms assume a simple nature of the script in the document and do not give good results on documents with complex scripts. We also emphasize that the segmentation algorithms can be improved by using context specific information along with visual information to perform segmentation. The script specific models can provide the *apriori* information for the segmentation algorithms to segment the documents with complex scripts.

References

[1] A. Antonacopoulos, D. Bridson, and B. Gatos. Page segmentation competition. In *ICDAR*, pages 75–79, 2005.

[2] T. M. Breuel. Two geometric algorithms for layout analysis. In *DAS: Proceedings of the International Workshop on Document Analysis Systems V*, pages 188–199, 2002.

[3] R. Cattoni, T. Coianiz, S. Messelodi, and C. Modena. Geometric layout analysis techniques for document image understanding: a review. In *IRST, Technical Report*, 1998.

[4] H.S.Baird, S.E.Jones, and S.J.Fortune. Image segmentation by shape-directed covers. In *Proceedings of ICPR*, pages 820–825, June 1990.

[5] K. S. Sesh Kumar, Anoop M. Namboodiri, and C. V. Jawahar. Learning segmentation of documents with complex scripts. In *ICVGIP*, pages 749–760, 2006.

[6] J. Kanai, S. Rice, T. Nartker, and G. Nagy. Automated Evaluation of OCR Zoning. *IEEE Transaction PAMI*, 17:86–90, 1995.

[7] T. Kanungo, R. M. Haralick, W. Stuezle, H. S. Baird, and D. Madigan. A statistical, nonparametric methodology for document degradation model validation. *IEEE Trans. PAMI*, 22(11):1209–1223, 2000.

[8] K. Kise, A. Sato, and M. Iwata. Segmentation of Page Images Using the Area Voronoi Diagram. *Computer Vision and Image Understanding*, 70:370–382, 1998.

[9] S. Kumar, N. Khanna, S. Chaudhury, and S. D. Joshi. Locating text in images using matched wavelets. In *ICDAR*, pages 595–599, 2005.

[10] J. Liang, I. Phillips, and R. Haralick. Performance Evaluation of Document Layout Analysis Algorithms on the UW Data Set. *Proceedings of SPIE Conference on Document Recognition*, 3027(IV):149–160, Feb 1997.

[11] S. Mao and T. Kanungo. Emperical performance evaluation methodology and its application to page segmentation algorithms. *IEEE Trans. PAMI*, 23(3):242–256, 2001.

[12] S. Mao, A. Rosenfeld, and T. Kanungo. Document structure analysis algorithms: A literature survey. In *Proceedings of SPIE Electronic Imaging*, 2003.

[13] G. Nagy, S. Seth, and M. Viswanathan. A Prototype Document Image Analysis System for Technical Journals. *Computer*, 25:10–22, 1992.

[14] L. O'Gorman. The Document Spectrum for Page Layout Analysis. *IEEE Trans. PAMI*, 15:1162–1173, 1993.

[15] U. Pal and B. B. Chaudhuri. Script line separation from indian multi-script documents. In *ICDAR*, pages 406–410, 1999.

[16] S. Randriamasy, L. Vincent, and B. Wittner. An Automatic Benchmarking Scheme for Page Segmentation. *Proceedings of SPIE Conference on Document Recognition*, 2181:217–230, Feb 1994.

[17] F. Shafait, D. Keysers, and T. M. Breuel. Performance comparison of six algorithms for page segmentation. In *Document Analysis Systems*, pages 368–379, 2006.

[18] F. Shafait, A. ul Hasan, D. Keysers, and T. M. Breuel. Layout analysis of urdu document images. In *10th IEEE International Multi-topic Conference (INMIC 2006)*, Dec 2006.

[19] K. Y. Wong, R. G. Casey, and F. M. Wahl. Document Analysis System. *IBM Journal of Research and Development*, 26(6):647–656, Nov. 1982.