Pixel-Accurate Representation and Evaluation of Page Segmentation in Document Images

Faisal Shafait Daniel Keysers Thomas M. Breuel
Image Understanding and Pattern Recognition (IUPR) Research Group
German Research Center for Artificial Intelligence (DFKI)
and Technical University of Kaiserslautern
D-67663 Kaiserslautern, Germany
{faisal, keysers, tmb}@iupr.net

Abstract

This paper presents a new representation and evaluation procedure of page segmentation algorithms and analyzes six widely-used layout analysis algorithms using the procedure. The method permits a detailed analysis of the behavior of page segmentation algorithms in terms of over- and undersegmentation at different layout levels, as well as determination of the geometric accuracy of the segmentation. The representation of document layouts relies on labeling each pixel according to its function in the overall segmentation, permitting pixel-accurate representation of layout information of arbitrary layouts and allowing background pixels to be classified as "don't care". Our representations can be encoded easily in standard color image formats like PNG, permitting easy interchange of segmentation results and ground truth.

1. Introduction

Page segmentation is the step in document image analysis that divides the document image into homogeneous zones, each consisting of only one physical layout structure (e.g. text, graphics, or pictures). Over the last decades, several page segmentation algorithms have been proposed in the literature (for a literature survey, please refer to [6, 13]). However, there is no widely accepted way of representing page segmentations. The Document Attribute Format Specification was developed with the intention to be used as a standard for the representation of document images and their partial interpretations during document decomposition [7]. However, it did not come to widespread use and other representations based on XML have emerged [1, 8]. A common problem with these approaches is that they need specialized software to view the files representing the page segmentation, thereby limiting their portability. We present a new way of representing layout information by embedding it in the color channels of a document image. Such a representation allows convenient interchange of ground truth and segmentation results in terms of standard image formats. A similar approach for representing handwriting segmentation was presented in [3].

The problem of automatic evaluation of page segmentation algorithms is becoming an important issue. Recent page segmentation competition [1] addresses the need of comparative performance evaluation under realistic circumstances. Several methods have been proposed for the evaluation of page segmentation algorithms [11, 12]. Major problems arise due to the lack of a common data set, a wide diversity of objectives, and inconsistencies in the use of document models. This makes the comparison of different page segmentation algorithms a difficult task. In addition, the evaluation methodology should be able to analyze different aspects of the segmentation algorithm, and give us a deeper insight into the behavior of the algorithm. We present a performance metric for evaluating page segmentation and analyze six well-known algorithms on that basis. The algorithms evaluated are X-Y cut [14], the smearing algorithm [17], whitespace analysis [2], Docstrum [15], the Voronoi-diagram based approach [10], and the constrained text-line finding algorithm [4].

A description of the color-based representation of segmentation in document images is given in Section 2, followed by the error metrics definitions in Section 3. Section 4 describes the experiments performed and results obtained, and the conclusion is presented in Section 5.

2. Representation of page segments

Consider a document image decomposed into N homogeneous zones $Z_i, i = 1, \ldots, N$. Then, the document segmentation can be represented as an image in which each foreground pixel is assigned as its value the index of the segment Z_i to which it belongs. In practice, the pixel-based representation of page segmentation can be implemented as 24-bit RGB color images. This enables the use of up to $N = 2^{24} - 1$ labels, which will be sufficient for virtually all images that are of interest. A particular color can be assigned to the page background (e.g. 0xffffff). This representation of page segmentation is particularly convenient because it is independent of the zone shape and it can be saved and exchanged using any lossless color image format.

3. Performance metric

Consider that we are given two segmentations in image form, the hypothesized segmentation H, and the ground truth G. The noise pixels in G are represented by a unique color (e.g. 0x000000). The images representing these segmentations should have the same dimensions. In order to compare the quality of a hypothesized segmentation against a ground truth segmentation, we can compute a weighted bipartite graph called *pixel-correspondence graph* [3] as follows. We index the distinct values that pixels in H and G assume. The nodes in the two components represent the indices of these values. Since each segment has a unique color, each node represents a unique segment (either in H or in G). An edge is constructed between two nodes such that the weight of the edge equals the number of foreground pixels in the intersection of the regions covered by the two segments represented by these nodes. If their corresponding segments do not overlap in H and G, no edge is needed.

If the hypothesized segmentation H agrees perfectly with the ground truth segmentation G, then the pixel-correspondence graph will be a perfect matching. That is, each node in both component of the graph has exactly one edge incident on it. If there are differences between the two segmentations, then the graph will not be a perfect matching. Instead, a node representing a segmentation in H or G may have multiple edges.

If P be the total number of pixels corresponding to one node (segment), M be the number of edges incident to that node, and $w_i, i = 1, 2, \ldots, M$ be the weight associated with each edge, then $P = \sum_{i=1}^{M} w_i$. For each node on either component of the graph, w_i/P gives the fraction of pixels overlapping with each of its corresponding nodes.

An edge between two nodes is considered *significant* if $w_i/P \geq t_r$ or $w_i \geq t_a$, where t_r is a relative threshold and t_a is an absolute threshold. The use of t_r allows a tolerance in the evaluation by ignoring fractional overlaps less than t_r. In practice, $t_r = 0.1$ is usually a good choice. However, if a segmentation algorithm completely fails and gives the whole page as one segment, regions containing less than 10% of the foreground pixels may get ignored. Therefore,

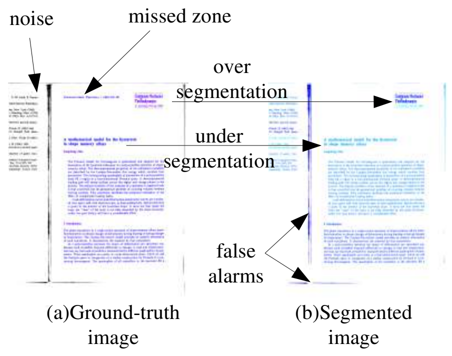

Figure 1. Example image to illustrate different error metrics.

an absolute threshold t_a is used to ensure that overlaps of more than t_a pixels are not ignored. We used $t_a = 500$ pixels for zone-level evaluation and $t_a = 100$ pixels for textline-level evaluation.

For a given node in the ground truth segmentation G, if there are more than one significant edge from the hypothesized segmentation H, then the node is considered *oversegmented*. Similarly, if a node in the hypothesized segmentation H has more than one significant edge to the ground truth segmentation G, then the node is considered *undersegmented*. Using these definitions, we can introduce the following metrics for evaluating a page segmentation algorithm.

Total oversegmentations (T_o): the total number of significant edges that ground truth components have, minus the number of ground truth components

Total undersegmentations (T_u): the total number of significant edges that segmentation components have, minus the number of segmentation components

Oversegmented components (C_o): the number of ground truth components having more than one significant edge.

Undersegmented components (C_u): the number of segmentation components having more than one significant edge.

Missed components (C_m): the number of ground truth components that matched the background in the hypothesized segmentation.

False alarms (C_f): Number of components in hypothesized segmentation that matched the background or noise in the ground truth segmentation.

4. Experiments and Results

Based on the performance metrics defined in Section 3, we evaluated the performance of six algorithms for page segmentations, namely, X-Y cut [14], the smearing algorithm [17], whitespace analysis [2], Docstrum [15], the

Table 1. **Different types of errors made by each algorithm on original zone-level groundtruth. Each text paragraph is considered a seperate text zone. The column labels are: total oversegmentations (T_o), total undersegmentations (T_u), oversegmented components (C_o), undersegmented components (C_u), missed components (C_m), false alarms (C_f)**

Algorithm	Groundtruth zones	Segmented zones	T_o	T_u	C_o	C_u	C_m	C_f
X-Y cut	24302	15035	3625	13579	1220	4212	384	6425
Whitespace	24302	19473	5227	10686	2275	4318	134	8552
Docstrum	24302	41605	12617	7453	2636	3516	342	18039
Voronoi	24302	41478	12333	8578	2779	3889	441	10348

Table 2. **Different types of errors made by each algorithm on textline-level ground truth. For a key to column labels please refer to Table 1.**

Algorithm	Ground truth zones	Segmented zones	T_o	T_u	C_o	C_u	C_m	C_f
Textline	105379	102979	1125	1433	624	1053	2145	29535
Smearing	105379	103838	3248	1312	1574	901	4121	40390

Voronoi-diagram based approach [10], and the constrained text-line finding algorithm [4]. The evaluation of the algorithms was done on the University of Washington III (UW-III) database [9]. The database consists of 1600 English document images with manually edited ground-truth of entity bounding boxes. These bounding boxes enclose text and non-text zones, text-lines, and words. We used all 1600 images for the evaluation. For each algorithm, parameters optimized for UW-III dataset were used as specified in [16]. The parameters for the X-Y cut, whitespace analysis, Docstrum, and Voronoi-diagram-based algorithms were tuned to find text zones. Hence, they were evaluated on zone-level ground truth with the results given in Table 1. The smearing, and the constrained textline finding algorithms locate textlines in the given image. So they are evaluated on textline-level ground truth with the results given in Table 2.

A problem with the text-zone level ground truth, in the UW-III dataset, is that a single paragraph is considered one text zone. Hence, two consecutive paragraphs on the same page make two different zones. In many documents, the segmentation of text columns into paragraphs is indicated by indentation rather than spacing. Determining paragraphs from indentations is usually a separate processing step. Therefore, an evaluation based on paragraph-level ground truth may not correctly reflect the performance of a page segmentation algorithm by giving more undersegmentation errors than the algorithm actually made.

We modified the ground truth for UW-III to get text-zones instead of paragraphs. For this purpose, we first specified a partial order of the text paragraphs based on their spatial relationships, and then used a topological sorting algorithm to find the reading order as in [5]. Then the bounding boxes of two consecutive paragraphs in the reading order were merged if their start and end positions along the horizontal direction are within 5 pixels of each other. These modified text-zones were used to evaluate the page segmentation algorithms, with the results as shown in Table 3.

Based on the results in Tables 2 and 3, we can make the following observations about each algorithm.

- The X-Y cut algorithm fails in the presence of noise and tends to take the whole page as one segment. This results in many undersegmentation errors.
- The whitespace algorithm is sensitive to the stopping rule. Early stopping results in a higher number of undersegmentation errors, late stopping results in more oversegmentation errors.
- In the Voronoi and Docstrum algorithms, the inter-character and inter-line spacings are estimated from the document image. Hence spacing variations due to different font sizes and styles within one page result in oversegmentation errors in both algorithms. For instance, in many cases, they fail to estimate the inter-line distance correctly, and hence split the zones into individual textlines, resulting in a large number of oversegmentation errors. The number of segmented zones for these two algorithms is more than double the number of zones in the ground truth.
- The smearing algorithm classifies text-lines merged with noise blocks as non-text, resulting in a large number of missed errors.
- The major part of the errors made by the constrained text-line finding algorithm are missed errors. Single digit page numbers are missed by the text-line finding

Table 3. Different types of errors made by each algorithm on modified zone-level ground truth. For a key to column labels please refer to table 1.

Algorithm	Ground truth zones	Segmented zones	T_o	T_u	C_o	C_u	C_m	C_f
X-Y cut	19960	14908	4239	10031	1748	3768	358	6570
Whitespace	19960	19501	6341	7716	2974	3711	114	8658
Docstrum	19960	41376	13953	4741	3787	2776	249	18223
Voronoi	19960	41155	13410	5705	3722	3240	349	10675

algorithm, because it requires at least two connected components to form a line. In some cases, the characters from two consecutive lines are merged. Hence, the bounding box of the lower textline spans across both textlines, resulting in both oversegmentation and undersegmentation errors.

5 Conclusion

We presented an approach for evaluating page-segmentation algorithms using color-based representation. The color-based representation of segmentation is independent of zone shape, and it can be saved and exchanged using any lossless color image format. Instead of using a single score for the performance of each algorithm, different aspects of the algorithms are evaluated separately. In general, the Docstrum and the Voronoi algorithms work better than X-Y cut and whitespace algorithms for page segmentation. For textline extraction, the constrained textline finding algorithm has better performance than the smearing algorithm. Depending on the target application, different error metrics may be weighted according to their significance in that application. However, for a direct comparison, an important next step will be to relate the measures that we developed in a formal way to page segmentation performance.

Acknowledgments

This work was partially funded by the BMBF (German Federal Ministry of Education and Research), project IPeT (01 IW D03).

References

[1] A. Antonacopoulos, B. Gatos, and D. Bridson. ICDAR 2005 page segmentation competition. In *Proc. ICDAR*, pages 75–80, Seoul, Korea, 2005.

[2] H. S. Baird. Background structure in document images. In H. Bunke, P. Wang, and H. S. Baird, editors, *Document Image Analysis*, pages 17–34. World Scientific, Singapore, 1994.

[3] T. M. Breuel. Representations and metrics for off-line handwriting segmentation. In *8th International Workshop on Frontiers in Handwriting Recognition*, pages 428–433, Ontario, Canada, Aug. 2002.

[4] T. M. Breuel. Two geometric algorithms for layout analysis. In *Proc. DAS*, pages 188–199, Princeton, NY, Aug. 2002.

[5] T. M. Breuel. High performance document layout analysis. In *Symposium on Document Image Understanding Technology, Greenbelt, MD*, 2003.

[6] R. Cattoni, T. Coianiz, S. Messelodi, and C. M. Modena. Geometric layout analysis techniques for document image understanding: a review. Technical report, IRST, Trento, Italy, 1998.

[7] D. Dori et. al. The representation of document structure: A generic object-process analysis. In H. Bunke and P. Wang, editors, *Handbook of character recognition and document image analysis*, pages 421–456. World Scientific, Singapore, 1997.

[8] G. Ford and D. Thoma. Ground truth data for document image analysis. In *Proceedings of 2003 Symposium on Document Image Understanding and Technology*, pages 199–205, Greenbelt, MD, April 2003.

[9] I. Guyon, R. M. Haralick, J. J. Hull, and I. T. Phillips. Data sets for OCR and document image understanding research. In H. Bunke and P. Wang, editors, *Handbook of character recognition and document image analysis*, pages 779–799. World Scientific, Singapore, 1997.

[10] K. Kise, A. Sato, and M. Iwata. Segmentation of page images using the area Voronoi diagram. *Computer Vision and Image Understanding*, 70(3):370–382, June 1998.

[11] J. Liang, I. T. Phillips, and R. M. Haralick. Performance evaluation of document structure extraction algorithms. *Computer Vision and Image Understanding*, 84:144–159, 2001.

[12] S. Mao and T. Kanungo. Empirical performance evaluation methodology and its application to page segmentation algorithms. *IEEE TPAMI*, 23(3):242–256, March 2001.

[13] S. Mao, A. Rosenfeld, and T. Kanungo. Document structure analysis algorithms: a literature survey. *Proc. SPIE Electronic Imaging*, 5010:197–207, Jan. 2003.

[14] G. Nagy, S. Seth, and M. Viswanathan. A prototype document image analysis system for technical journals. *Computer*, 7(25):10–22, 1992.

[15] L. O'Gorman. The document spectrum for page layout analysis. *IEEE TPAMI*, 15(11):1162–1173, Nov. 1993.

[16] F. Shafait, D. Keysers, and T. M. Breuel. Performance comparison of six algorithms for page segmentation. In *Proc. DAS*, pages 368–379, Nelson, New Zealand, Feb. 2006.

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