Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

An adaptive over-split and merge algorithm for page segmentation[☆]

Ha Dai-Ton[a,*], Nguyen Duc-Dung[b], Le Duc-Hieu[b]

[a] *Ha Long High School for Gifted Student, Ha Long City, Vietnam*
[b] *Institute of Information Technology, Vietnam Academy of Science and Technology, 18 Hoang Quoc Viet, Hanoi, Vietnam*

ARTICLE INFO

Article history:
Received 28 September 2015
Available online 25 June 2016

Keywords:
Document analysis and recognition
Page segmentation
OCR

ABSTRACT

Page segmentation is a key step in building a document recognition system. Variation in character font sizes, narrow spacing between text blocks, and complicated structure are main causes of the most common over-segmentation and under-segmentation errors. We propose an adaptive over-split and merge algorithm to reduce simultaneously these types of error. The document image is firstly over-split into text blocks, even text lines. These text blocks are then considered to merge into text regions using a new adaptive thresholding method. Local context analysis uses a set of text line separators to split homogeneous text regions of similar font size and close text blocks into paragraphs. Experiments on the ICDAR2009 and UW-III benchmarking datasets show the effectiveness of the proposed algorithm in reducing both the under and over-segmentation errors and boost the performance significantly when comparing with popular page segmentation algorithms.

1. Introduction

Document layout analysis is one of the main components in an OCR system. The task of document layout analysis includes automatically detecting zones on a document image (physical layout analysis – page segmentation) and classifying them into different regions such as: text, images, tables, header, footer, etc. (logical layout analysis). The results of page segmentation are used as an input to the process of recognition and automatic data entry of image processing systems in general.

Compared with the logical layout analysis, the physical layout analysis has attracted more attention of researchers because of its diverse and complex layout of different types of document. Not only the specific types of document (books, newspapers, magazines, reports, etc.) but also other factors of a page such as editors and font size, layout, alignment constraints, etc affect detection and segmentation accuracy of the algorithm.

Based on the order of processing, page segmentation algorithms are primarily divided into three categories: bottom-up, top-down and hybrid. Bottom-up algorithms are both the oldest, e.g. [17] and more recently published, e.g. [7,8,14], algorithms. They classify small parts of the image (pixels, groups of pixels, or connected components), and gather those of the same type together to form regions. The key advantage of bottom-up algorithms is that they can handle arbitrarily shaped regions with ease (rectangular or non-rectangular). However, the sensitivity of the measure used to form higher-level entities is the main disadvantage of this approach; this often leads to over-segmentation error in the page with many changes in font sizes and styles, especially the titles. Top-down algorithms, e.g. [5,13] cut the image recursively in vertical and horizontal directions along with white-spaces that are expected to be column boundaries or paragraph boundaries. Although top-down algorithms have low computation complexity and good separation result on images with rectangular layout, they are not really able to handle the variety of formats that occur in many magazine pages, such as non-rectangular regions and cross-column headings that blend seamlessly into the columns below. This leads to under-segmentation error.

The third type of algorithm, e.g. [16], is based on bottom-up method to find delimiters, such as rectangular white-spaces, tab-stops, etc. These delimiters are then used to infer a top-down layout of document image. After that, the algorithms use a bottom-up method and top-down layout to detect text regions. Therefore, hybrid algorithms can overcome over-segmentation error caused by bottom-up algorithms and perform better than top-down algorithms when dealing with non-rectangular text regions. However, it is not trivial to detect delimiters exactly by many reasons, e.g. text regions are very close to each other, text regions are not left or right aligned, spaces of connected components are large, which lead to misidentified delimiters or missed delimiters. So, the results of the algorithms often contain both

☆ This paper has been recommended for acceptance by Jose Ruiz Jose Ruiz-Shulcloper.
* Corresponding author. Tel.: +84 23 498 1188, +84 1234981188.
E-mail address: hadaiton83@gmail.com (H. Dai-Ton).

http://dx.doi.org/10.1016/j.patrec.2016.06.011
0167-8655/© 2016 Elsevier B.V. All rights reserved.

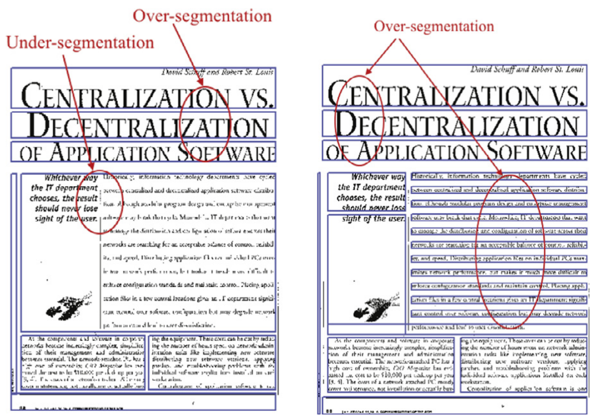

Fig. 1. Illustration of under-segmentation and over-segmentation errors.

over-segmentation and under-segmentation errors as illustrated in Fig. 1. In short, the over-segmentation and under-segmentation errors are the most frequent types and they are, in fact, not easy to overcome. Fixing the over-segmentation error usually leads to the under-segmentation error, and vice versa.

In this paper, we present an Adaptive Over-Split and Merge (AOSM) algorithm for overcoming both the over- and under-segmentation errors in page segmentation problem. AOSM firstly over-segments page image using a set of white-spaces covering the whole document background. It then groups over-segmented text regions using adaptive parameters. Finally, local context analysis sub-divides (under-segmented) text regions into paragraphs. Experimental results on the ICDAR2009 page segmentation competition and the UW-III datasets show that AOSM reduces both over-segmentation and under-segmentation errors significantly, thus boost the performance of page segmentation when comparing with the state-of-the-art algorithms.

The rest of this paper is organized as follows. In Section 2 we describe in detailed the AOSM algorithm. Section 3 gives experiment results and analysis on two benchmark datasets ICDAR2009 and UW-III. Finally, conclusions and discussions are given in Section 4.

2. Adaptive over-split and merge page segmentation

Fig. 2 outlines two stages and main processing steps of the proposed AOSM algorithm. The first stage aims at quickly dividing page images into regions and sub-regions. The second stage groups text regions using adaptive thresholds and then once again separate text blocks into paragraphs using local context analysis. Details of the main steps are presented in the following sub-sections.

2.1. Phase 1: Over-segmentation

Instead of using tab-stops as delimiters, e.g. [16], AOSM uses all available rectangular white-spaces covering background of document as delimiters. It is to eliminate under-segmentation error of the conventional top-down methods.

2.1.1. Connected component filtering

The morphological processing [4] firstly detects vertical, horizontal lines and image regions. The detected elements are subtracted from the image before passing to connected components analysis. The connected components are then filtered by their size into small (likely as noise), large (likely as image halftone) and the rest are medium (likely as text - CCs).

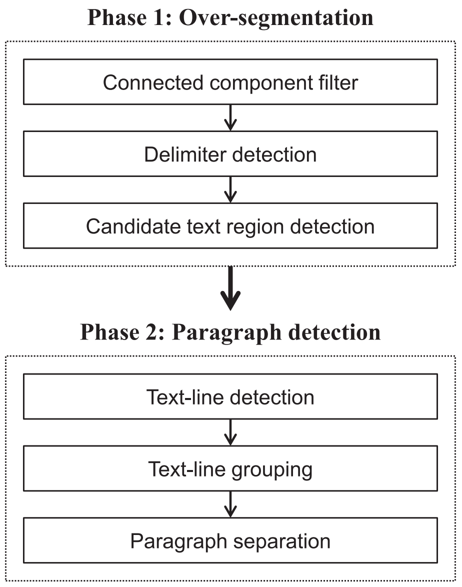
Phase 1: Over-segmentation
Connected component filter
Delimiter detection
Candidate text region detection
Phase 2: Paragraph detection
Text-line detection
Text-line grouping
Paragraph separation

Fig. 2. The process chain of AOSM algorithm.

2.1.2. Delimiter detection

When a document is written and laid out by a word processor or a professional publishing system, text regions are usually bounded and thus differentiated from each other by means of delimiters. The delimiters can either be long horizontal/vertical line segments (dubbed solid separator), physical delimiters (distance of CCs), large elongated empty areas (dubbed whites-paces) or the chain of alignment connected components (tab-stops), etc.

There have been different approaches for delimiter detection in which analyzing the structure of the white background is one of the common methods to detect delimiter, e.g. the WhiteSpace algorithm [5]. It is the fact that white space is a generic layout delimiter and background structure is simpler than those of the foreground. However, the segmentation result of the WhiteSpace algorithm is rather sensitive to the stopping rule (based on the number of delimiters). Early stopping results in a higher number of under-segmentation errors and late stopping results in more over-segmentation errors [15]. To overcome this limitation, AOSM uses the set of all white spaces covering background document as delimiters (Fig. 3). This policy solves not only the delimiter detection but also the under-segmentation problem.

2.1.3. Candidate text region detection

The white spaces discovered in the previous section are removed from page image and the remaining candidate text regions are detected easily (Fig. 4). In the next section, candidate of text lines within each separated region will be detected efficiently and the potential of joining text lines in two neighboring regions is mostly eliminated.

2.2. Phase 2: Paragraph detection

The output of Phase 1 intentionally consists of over-segmented text regions and the task of Phase 2 is to detect and fix these errors.

2.2.1. Text-line extraction and grouping

CCs in each detected candidate text region are scanned from left to right and from top to bottom to form text-lines. After that, two text-lines $line_i$ and $line_j$ (belonging to two neighboring text regions) are considered to group into one region if they satisfy