Simultaneous Word Segmentation from Document Images Using Recursive Morphological Closing Transform

Su Chen and Robert M. Haralick

Department of Electrical Engineering
University of Washington
Seattle, WA 98195

Ihsin T. Phillips

Department of Computer Science
Seattle University
Seattle, WA 98122

Abstract

This paper describes a word segmentation algorithm which is based on the recursive morphological closing transform. The algorithm is trainable for any given document image population and is capable of detecting words on a document image simultaneously.

We describe an experimental protocol to train and evaluate our word segmentation algorithm based on a set of layout ground-truthed document images. We also discussed a method to compare two sets of word bounding boxes — one from the ground truth and the other from the output of the word segmentation algorithm, and compute the numbers of miss, false, correct, splitting, merging and spurious detections.

The experimental results demonstrate that under the optimal algorithm parameter settings, the correct word detection percentage is about 95% *on both the training and testing image populations. If this includes the splitting and merging detections, the detection percentage is about* 99.4%.

1 Introduction

The document layout analysis identifies various objects of interest on a document image and describes their spatial relations. Earlier work on document layout analysis can be categorically divided into two groups. One group employs the top-down or model-driven approach [1]. It starts at the global image level and successively decomposes the image into smaller regions. Each region corresponds to either characters, words, text lines, paragraphs text columns or non-textual regions. The other group adopts the bottom-up or data-driven approach [2]. It starts by synthesizing evidence at the black-and-white pixel level and then merges pixels into characters, characters into words, words into lines, lines into paragraphs and paragraphs into columns, etc., until the whole document is completely labeled.

The main problems associated with these earlier techniques are that they were developed on a trial-and-error basis and although they provide illustrative results, hardly any have been tested on significant sized data sets. In addition, most of them do not give any explicit quantitative performance measures of their systems.

In this paper, we approach the problem in a systematic way. Section 2 provides the general statement of the word segmentation problem. Section 3 shows how the problem relates into a classical pixel classification problem. Then we describes a word segmentation algorithm using the recursive morphological closing transform (RCT) and a MAP classifier. Section 5 discusses an experimental protocol to train and evaluate the algorithm based on a set of layout ground-truthed document images [6]. Finally, Section 5, summarizes our experimental results.

2 Problem statement

Let I denote a bi-level document image. Let Σ denote the set of word bounding boxes on I. The problem of the word segmentation can be simply formulated as follows:

Word Segmentation Problem:
Given a document image I. *Find* Σ *to maximize the conditional probability* $P(\Sigma \mid I)$.

3 Word segmentation using RCT

The set of word bounding boxes Σ provides a delineation of two types of regions in the image I, namely the word and non-word regions. We define a pixel as a word pixel if and only if it is on or inside a word bounding box in Σ. Then, a word region consists solely of word pixels; whereas a non-word region is composed of only non-word pixels.

Associated with each pixel $x \in I$, there are a random observation vector $\mathcal{Y} = y$ that characterizes the image shapes around x and a label $\mathcal{L} = l$ that indicates whether x is a word pixel (denoted by $\mathcal{L} = 1$) or a non-word pixel (denoted by $\mathcal{L} = 0$). Let $\mathcal{Y}_I$ and $\mathcal{L}_I$ represent images of observation vectors and labels, respectively. Then, we can re-formulate the word segmentation problem as finding $\mathcal{L}_I$ to maximize $P(\mathcal{L}_I \mid \mathcal{Y}_I)$ and then computing the bounding boxes for the connected regions in $\mathcal{L}_I$.

In general, we would model $\mathcal{Y}_I$ as a Markov random field. Its solution usually demands iterative schemes and involves much computation. In the following, we will describe a very simple and fast solution.

0-8186-7128-9/95 $4.00 © 1995 IEEE

Proceedings of the Third International Conference on Document Analysis and Recognition (ICDAR '95)
0-8186-7128-9/95 $10.00 © 1995 IEEE

Step 1: sub-sampling

Assume that our input document images are scan-digitized at a spatial resolution of 300dpi. For a standard $11'' \times 8.5''$ page, it is equivalent to an input document image size of 3300×2550. To process such an image, it will take more memory and processing time. Hence, we sub-sample the input image to 150dpi. Figure 1 (a) illustrates the sub-sampled image.

Step 2: word block detection

Our computation of the observation vector $\mathcal{Y} = y$ is based on the recursive closing transform [3]. The recursive closing transform provides a powerful tool to extract shape information in the image background (white-space), such as the pattern spectrum.

Let $K_1, K_2, \cdots, K_n$ denote n structuring elements. Let $y_1 = CT[I, K_1](x)$, $y_2 = CT[I, K_2](x)$, $\cdots$, $y_n = CT[I, K_n](x)$ denote the values of the recursive closing transform at pixel $x \in I$ with respect to the structuring elements $K_1, K_2, \cdots, K_n$. Let $y = (y_1, y_2, \cdots, y_n)$. In our current configuration, we choose $n = 3$ and K_1 is the horizontal 1×2 structuring element, K_2 is the vertical 2×1 structuring element, K_3 is the 2×2 square structuring element.

We assume that the observation vectors from different pixel locations are independent, i.e.

$$P(\mathcal{L}_I \mid \mathcal{Y}_I) = \prod_{x \in I} P(\mathcal{L} \mid \mathcal{Y}).$$

To perform the optimization, we assign a posterior probability $P(\mathcal{L} = 1 \mid \mathcal{Y} = y)$ to each pixel for being a word pixel. Then the posterior probability for each pixel to be a non-word pixel is equal to $P(\mathcal{L} = 0 \mid \mathcal{Y} = y) = 1 - P(\mathcal{L} = 1 \mid \mathcal{Y} = y)$. The output is a posterior probability map image. If the posterior probability map image is thresholded at $T_p = 0.5$, i.e. pixels that have values greater than or equal to T_p have binary one output values, we obtain the maximum a posteriori (MAP) solution. But in general, we could choose T_p between 0.5 and 1.0. Figure 1 (b) illustrates the probability map image. Figure 1 (c) illustrates the detected word label image, where $T_p = 0.96$.

Step 3: word bounding box extraction

Each detected word block is modeled as a 8-connected connected component. A connected component labeling procedure is performed on the binary word block image. The bounding box of each of the connected components is calculated. Figure 1 (d) illustrates the sub-sampled image overlaid with the extracted word bounding boxes.

Step 4: hypothesis test on word height

The presence of the character ascenders and descenders sometimes causes the merging of many word blocks from two or more adjacent text lines into one big block. In order to automatically detect such cases and consequently split the merged word blocks into their corresponding correct words, we developed a simple post-processing procedure to perform a hypothesis test on the height of the word blocks and test if further divisions are needed.

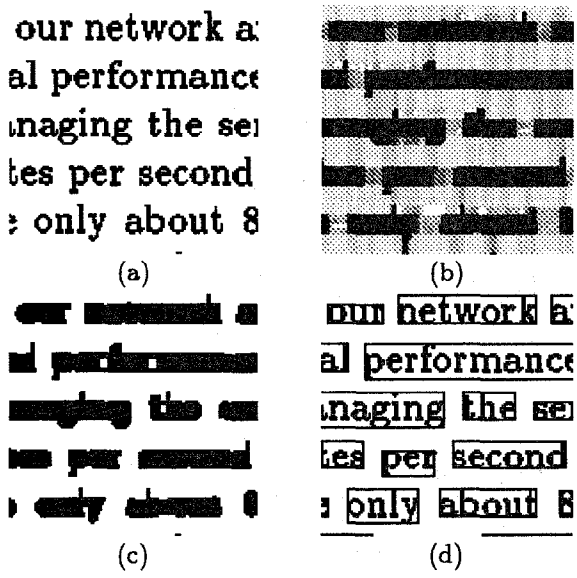

Figure 1: Illustrates the word segmentation process. (a) sub-sampled 150dpi image; (b) posterior probability map image; (c) thresholded word block image; (d) word bounding boxes.

Let W_h denote the dominant word height of a given document image population. Then the procedure hypothesizes that all the detected word blocks whose heights exceed βW_h could be split further, where β is a real constant and has a default value of $\beta = 2.0$. For each word block which is hypothesized to be divided further, the algorithm will verify it by computing cut points in the projection profile of the posterior probability map image along the height direction within the bounding box of the dubious word block. The cut points are defined as the local minimums of the profile in a neighborhood of size W_h and whose values are less than or equal to a cut-point threshold.

4 Experimental protocol

In the last section, we outlined our word segmentation algorithm. The algorithm is not yet fully operational because the posterior probability $P(\mathcal{L} = 1 \mid \mathcal{Y} = y)$ has to be estimated through the experiment.

4.1 Posterior probability estimation

The estimation of the posterior probability distribution is based on the 168 synthetic document images from the UW-I English document image database. We reported an automatic procedure to create the ground

Proceedings of the Third International Conference on Document Analysis and Recognition (ICDAR '95)
0-8186-7128-9/95 $10.00 © 1995 IEEE

truth word bounding boxes for these images in [6]. Then given a set of ground truth word bounding boxes, we generate a so-called word mask image for each of the training document images. The word mask image is bi-level and has a binary one pixel if and only if the pixel is a word pixel.

Each document image and its corresponding word mask image are then rotated at various degrees of $0°$, $\pm 0.2°$, $\pm 4°$, $\pm 0.6°$, using a nearest neighbor interpolation algorithm. This becomes a total training image population of $1176 = 168 \times 7$ images. Each image is of size 1650×1275.

We adopt a rather brute-force method to estimate the posterior probability $P(\mathcal{L} = 1 \mid \mathcal{Y} = y)$:

$$P(\mathcal{L} = 1 \mid \mathcal{Y} = y) = \frac{P(\mathcal{L} = 1, \mathcal{Y} = y)}{P(\mathcal{Y} = y)} = \frac{P(\mathcal{L} = 1, \mathcal{Y} = y)}{P(\mathcal{L} = 0, \mathcal{Y} = y) + P(\mathcal{L} = 1, \mathcal{Y} = y)}$$

The joint probability distributions can be substituted with the frequency counts $\#(\mathcal{L} = 0, \mathcal{Y} = y)$ and $\#(\mathcal{L} = 1, \mathcal{Y} = y)$. The counting processes are simplified in our case because the observation vectors $\mathcal{Y} = (y_1, y_2, y_3)$ are integer vectors and bounded within the 3-dimensional cube $[0, N] \times [0, N] \times [0, N]$, where N is the allowed maximum output integer value of the closing transform [3]. For word segmentation, we choose $N = 63$.

In this paper, we further assume that $P(\mathcal{L} = 1 \mid \mathcal{Y} = y)$ is symmetric with respect to the first two coordinates of $\mathcal{Y}$, i.e. $P[\mathcal{L} = 1 \mid \mathcal{Y} = (y_1, y_2, y_3)] = P[\mathcal{L} = 1 \mid \mathcal{Y} = (y_2, y_1, y_3)]$. This will permit the posterior probability distribution to characterize text words laid out in both the horizontal and the vertical directions. Therefore, we instead substitute the $P(\mathcal{L} = 0, \mathcal{Y} = y)$ with the frequency count $\#(\mathcal{L} = 0, \mathcal{Y} = (y_1, y_2, y_3)) + \#(\mathcal{L} = 0, \mathcal{Y} = (y_2, y_1, y_3))$ and the $P(\mathcal{L} = 1, \mathcal{Y} = y)$ with the frequency count $\#(\mathcal{L} = 1, \mathcal{Y} = (y_1, y_2, y_3)) + \#(\mathcal{L} = 1, \mathcal{Y} = (y_2, y_1, y_3))$.

4.2 Segmentation algorithm evaluation

The output of the word segmentation algorithm is a set of word bounding boxes. To evaluate its performance, we need to compare the output word bounding boxes with the ground truth word bounding boxes. Let $\mathcal{G} = \{G_1, G_2, \cdots, G_N\}$ represent the total of N ground truth word bounding boxes and let $\mathcal{D} = \{D_1, D_2, \cdots, D_M\}$ denote the total of M detected word bounding boxes from the word segmentation algorithm. The evaluation problem can be formally stated as follows:

Given two sets of bounding boxes $\mathcal{G}$ and $\mathcal{D}$. Establish the element mappings between the two sets and report the numbers of miss detections (1-0 mappings), false detections (0-1 mappings), correct detections (1-1 mappings) and splitting detections (1-m mappings), merging detections (m-1 mappings) and spurious detections (m-m mappings).

To establish the element mappings, we first define the similarity between two bounding boxes A and B, denoted by $s(A, B)$:

$$s(A, B) = \frac{Area(A \cap B)}{Area(A)}$$

where $A \cap B$ denotes the region where A and B overlap. The similarity defines the percentage area coverage of A by B.

Then based on the similarity measure, we define two mappings $g : \mathcal{G} \rightarrow \mathcal{D}$ and $d : \mathcal{D} \rightarrow \mathcal{G}$:

$$g(G_i) = \{D_j \in \mathcal{D} \mid G_i = arg \max_{X \in \mathcal{G}} s(D_j, X)\}$$

$$d(D_j) = \{G_i \in \mathcal{G} \mid D_j = arg \max_{X \in \mathcal{D}} s(G_i, X)\}$$

where $g(G_i)$ denotes the set of $D_j \in \mathcal{D}$ that has the highest percentage area coverage by G_i among all other boxes in $\mathcal{G}$. and $d(D_j)$ denotes the set of $G_i \in \mathcal{G}$ that has the highest percentage area coverage by D_j among all other boxes in $\mathcal{D}$. Therefore, we establish links from G_i to $g(G_i)$ and from D_j to $d(D_j)$.

Based on the two functions $g : \mathcal{G} \rightarrow \mathcal{D}$ and $d : \mathcal{D} \rightarrow \mathcal{G}$, we could establish mappings between the elements of $\mathcal{G}$ and $\mathcal{D}$. The rules are described as follows:

1. If there exists a G_i such that $s(G_i, D_j) = 0$ for all $j = 1, 2, \cdots, M$, then the G_i is counted as a miss detection (1-0 mapping).

2. If there exists a D_j such that $s(D_j, G_i) = 0$ for all $i = 1, 2, \cdots, N$, then the D_j is counted as a false detection.

3. There is a correct detection (1-1 mapping) between G_i and D_j if and only if $g(G_i) = \{D_j\}$ and $d(D_j) = \{G_i\}$.

4. There is a splitting detection (1-m mapping) between G_i and $\{D_{j_1}, D_{j_2}, \cdots, D_{j_m}\}$ if and only if, 1) $g(G_i) = \{D_{j_1}, D_{j_2}, \cdots, D_{j_m}\}$; 2) There exists one $D_0 \in g(G_i)$ such that $d(D_0) = \{G_i\}$ and for all $D \in g(G_i)$ but $D \neq D_0$, $d(D) = \emptyset$; 3) For all $D \notin g(G_i)$, $G_i \notin d(D)$.

5. There is a merging detection (m-1 mapping) between $\{G_{i_1}, G_{i_2}, \cdots, G_{i_m}\}$ and D_j if and only if, 1) $d(D_j) = \{G_{i_1}, G_{i_2}, \cdots, G_{i_m}\}$; 2) There exists one $G_0 \in d(D_j)$ such that $g(G_0) = \{D_j\}$ and for all $G \in d(D_j)$ but $G \neq G_0$, $g(G) = \emptyset$; 3) For all $G \notin d(D_j)$, $D_j \notin g(G)$.

6. Any other detections are counted as spurious detections (m-m mappings). □

Once the element mappings between $\mathcal{G}$ and $\mathcal{D}$ has been established, we could count the numbers of miss, false, correct, splitting, merging and spurious detections. Let N_{10}, N_{01} and N_{11} be the numbers of miss, false and correct detections, respectively. Let N^g_{1m}, N^g_{m1} and N^g_{mm} denote the numbers of words in the $\mathcal{G}$ that have the 1-m, m-1 and m-m mappings

Proceedings of the Third International Conference on Document Analysis and Recognition (ICDAR '95)
0-8186-7128-9/95 $10.00 © 1995 IEEE

with words in the $\mathcal{D}$. Similarly, let N^d_{1m}, N^d_{m1} and N^d_{mm} denote the numbers of words in the $\mathcal{D}$ that have the 1-m, m-1 and m-m mappings with words in the $\mathcal{G}$. Then the following relations satisfy: 1) $N = N_{10} + N_{11} + N^g_{1m} + N^g_{m1} + N^g_{mm}$; 2) $M = N_{01} + N_{11} + N^d_{1m} + N^d_{m1} + N^d_{mm}$; 3) $N^g_{1m} \leq N^d_{1m}$; 4) $N^g_{m1} \geq N^d_{m1}$.

The performance of the word segmentation algorithm can be measured through a goodness function. Let it be denoted as κ and be defined as:

$$\kappa = min(\kappa_1, \kappa_2)$$

where

$$\kappa_1 = (\gamma_{10}N_{10}+\gamma_{11}N_{11}+\gamma_{1m}N^g_{1m}+\gamma_{m1}N^g_{m1}+\gamma_{mm}N^g_{mm})/N$$

$$\kappa_2 = (\gamma_{01}N_{01}+\gamma_{11}N_{11}+\gamma_{1m}N^d_{1m}+\gamma_{m1}N^d_{m1}+\gamma_{mm}N^d_{mm})/M$$

and the γ_{10}, γ_{01}, γ_{11}, γ_{1m}, γ_{m1} and γ_{mm} are economic gain coefficients for the miss, false, correct, splitting, merging and spurious detections. The larger the goodness measure κ, the better the performance of the word segmentation algorithm. In the experiment, we choose the economic gain coefficients as in Table 1:

Table 1: Economic Gain Coefficients

γ_{10}	γ_{01}	γ_{11}	γ_{1m}	γ_{m1}	γ_{mm}
0.0	0.0	1.0	0.5	0.5	0.0

4.3 Optimal threshold determination

In the word segmentation algorithm, there is a threshold value T_p that needs to be computed on a per image basis. Therefore, it is necessary to develop an automatic procedure to predict the optimal threshold value on the fly. Our approach to this problem is to first determine the optimal threshold values for each of the training document images and then construct a regression function to predict the optimal threshold value given the histogram of the posterior probability map image [5].

Given an input document image, κ is a function of the threshold value T_p, i.e. $\kappa = \kappa(T_p)$. The optimal T_p is defined as the value that produces the best word segmentation goodness measure. Let T^{opt}_p denote the optimal threshold value. Then,

$$T^{opt}_p = arg\ [\max_{T_p \in [0,1]} \kappa(T_p)].$$

5 Experimental results

To determine the optimal performance of our word segmentation algorithm, we prepared a set of 96 testing document images and created their ground truth word bounding boxes using the procedure described in [6]. We also rotate each of the training document images and its corresponding ground truth word bounding boxes at various degrees of 0°, ±0.2°, ±4°, ±0.6°.

Under the optimal threshold settings ($T_p = T^{opt}_p$), of the 258328 ground truth words, 95.0667% of them are correctly detected. There are 1.6015% and 2.7573% of the words are split or merged, respectively. The total miss (0.3275%) and spurious (0.2470%) detections account for less than 0.6% of the total ground truth words. On the other hand, of the 258802 words detected by the algorithm, 94.8926% of them are correctly detected as the ground truth words. There are 3.5896% and 1.1441% of the detected words are derived from either split or merged ground truth words, respectively. The total false (0.1209%) and spurious (0.2527%) detections account for less than 0.4% of the total algorithm output.

6 Conclusion

We described a word segmentation algorithm that is able to detect all the words on a document image simultaneously. The algorithm is trainable to any given document image population and is not sensitive to text skews. It can handle a reasonable amount of text skews, where texts can be laid out in both the horizontal and the vertical directions at the same time. The algorithm is robust under subtractive noise and tolerant to some forms of additive noise.

References

[1] D. Wang and S.N. Srihari, "Classification of newspaper image blocks using texture analysis", *Computer Vision, Graphics and Image Processing*, 47:327-352, 1989.

[2] L.A. Fletcher and R. Kasturi, "A robust algorithm for text string separation from mixed text/graphics images", *IEEE Trans. on PAMI*, Vol. 10, No. 6, 1988, pp. 910-918.

[3] S. Chen and R.M. Haralick, "Recursive Erosion, Dilation, Opening and Closing Transforms", *IEEE Trans on Image Processing*, Vol.4, No. 3, March 1995.

[4] I.T. Phillips, S. Chen and R.M. Haralick, "English Document Database Standard", *ICDAR*, Japan, 1993.

[5] S. Chen and R.M. Haralick, "An automatic algorithm for text skew estimation in document images using recursive morphological transforms", *ICIP*, Austin, November, 1994.

[6] S. Chen, R.M. Haralick and I.T. Phillips, "Perfect document layout ground truth generation using DVI files and simultaneous text word detection from document images", *Proc. of the 4rd Annual Symposium on Document Analysis and Information Retrieval*, Las Vegas, April, 1995.

Proceedings of the Third International Conference on Document Analysis and Recognition (ICDAR '95)
0-8186-7128-9/95 $10.00 © 1995 IEEE