Handwritten Text Line Segmentation by Clustering with Distance Metric Learning

Fei Yin, Cheng-Lin Liu
National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences, P.O. Box 2728, Beijing 100190, P.R. China
E-Mail: {fyin, liucl}@nlpr.ia.ac.cn

Abstract

Separating text lines in handwritten documents remains a challenge because the text lines are often unevenly skewed and curved. In this paper, we propose a novel text line segmentation algorithm based on Minimal Spanning Tree (MST) clustering with distance metric learning. Given a distance metric, the connected components of document image are grouped into a tree structure. Text lines are extracted by dynamically cutting the edges of the tree using a new objective function. For avoiding artificial parameters and improving the segmentation accuracy, we design the distance metric by supervised learning. Experiments on handwritten Chinese documents demonstrate the superiority of the approach.

Keywords: Handwritten Text Line Segmentation, Clustering, Minimal Spanning Tree (MST), Distance Metric Learning.

1. Introduction

Document image analysis involves the tasks of text block segmentation, text line separation, character segmentation and recognition, and linguistic processing. Whereas the difficulty of machine-printed document analysis mainly lies in the complex layout structure and degraded image quality, handwritten document analysis is difficult mainly due to the irregularity of layout and character shapes originated from the variability of writing styles. For handwritten documents, text line segmentation and character segmentation-recognition are not solved though enormous efforts have been devoted to them.

Text line segmentation of handwritten documents is much more difficult than that for printed documents. Unlike that printed documents have approximately straight and parallel text lines, the lines in handwritten documents are often un-uniformly skewed and curved, and the inter-line spacing is usually not uniform. In many cases, the space between handwritten text lines is not obvious compared to the space between within-line characters, and some text lines may interfere with each other. Therefore, many text line detection techniques for printed documents, such as projection analysis and Hough transform, are not able to detect handwritten text lines.

Some efforts have been devoted to the difficult problem of handwritten text line segmentation [1-6]. The approaches can be roughly categorized into top-down and bottom-up ones. Top-down methods partition the document image recursively into text regions, text lines, and words/characters. Bottom-up methods group small units of image (pixels, connected components, characters, words, etc.) into text lines and then text regions. Bottom-up grouping can be viewed as a clustering process, which aggregates image components according to proximity and does not rely on the assumption of straight lines.

Both top-down and bottom-up methods have their disadvantages. Top-down methods do not perform well on curved and overlapping text lines. Bottom-up grouping is more complicated in computation than top-down partitioning, and its performance relies on some heuristic rules or artificial parameters, such as the between-component distance metric for clustering.

In this paper, we propose an effective bottom-up method for text line segmentation in unconstrained handwritten documents. Our approach is based on Minimal Spanning Tree (MST) clustering, and has no artificial parameter. The distance metric between connected components is designed by supervised learning. We compare this learning-based clustering method with that with artificially designed metric and show that supervised metric learning improves largely the accuracy of text line segmentation.

In the rest of this paper, Section 2 briefly reviews some related works; Section 3 gives an overall description of our clustering-based method and Section 4 is devoted to distance metric learning. Section 5 presents our experimental results and Section 6 concludes the paper.

2. Previous Works

Document structure is a hierarchy of text regions, text lines, words, characters and connected components. Text lines can be extracted by either top-down region
partitioning or bottom-up component aggregation. Some representative top-down and bottom-up segmentation methods are reviewed below.

The X-Y cut algorithm [7] is a projection-based top-down segmentation method but performs well only on printed documents because of the assumption of parallel text lines and large between-line gaps. Some researchers modified the projection-based method to deal with slightly curved text lines. To do this, the document image is partitioned into several vertical strips [3]. The text lines in each strip (assumed to be approximately straight) are extracted according to horizontal projection profiles and then connected with the lines of other strips by heuristic rules. Zahour et al. proposed a partial projection-based method combined with slant detection and partial contour tracing [2]. From a different viewpoint, several researchers proposed smearing-based top-down methods. Shi et al. use an adaptive local connectivity map (ALCM), in which the value of each pixel is the sum of all pixels in the original image within a specified horizontal distance [4]. After thresholding the smeared image, the connected components then represent probable regions of text lines. Kennard and Barrett use a similar method with slight extension [5] to deal with free-form handwritten documents. The recently proposed level set based method [6] is an effective top-down approach for unconstrained handwritten documents. On converting a binary image to gray-scaled, the level set method is exploited to determine the boundary between neighboring text lines. An obvious flaw of this algorithm is its high computation complexity.

The Docstrum method of O’Gorman [8] is typical of bottom-up grouping. It merges neighboring connected components using rules based on the geometric relationship between K nearest neighbor units, and performs well on printed documents as well as slightly curved handwritten documents. Likforman-Sulem and Faure [1] developed an iterative method based on perceptual grouping using three Gestalt criteria, namely, proximity, similarity and direction continuity, to group connected components. The grouping of components to text lines can be considered as a clustering problem, and has been treated using minimal spanning tree (MST) clustering [9][10]. The performance of clustering relies on the distance metric between components. The Hough transform algorithm has also been applied to handwritten text line detection [11][12], with the gravity centers or minima points of connected components as the points to be fitted, but needs a sophisticated post-processing procedure to extract the lines.

3. Clustering-Based Text Line Separation

In this section, we describe our overall approach of text line segmentation, the MST clustering algorithm, and the text line extraction procedure after clustering. The distance metric is specially treated in Section 4.

3.1. Overall approach

In a document image, each text line can be viewed as a cluster of stroke pixels or connected components (CCs). We prefer using the CCs as basic units of clustering because the CCs are easy to detect and the number of CCs is much smaller than that of stroke pixels. The minimal spanning tree (MST) algorithm is suitable for clustering the CCs into text lines, because it is computationally efficient and capable of detecting clusters with irregular boundaries. Two important issues in clustering are the distance metric between units and the criterion for determining the number of clusters.

To find a good metric for clustering CCs, we regard the important relationship of CCs that the distance between two neighboring components in the same text line should be smaller than that between different lines. The Euclidean distance does not meet with this requirement. We previously used a hand-crafted distance metric [13], which works fairly well but not sufficiently. We hereby design a better metric by supervised learning. By labeling some component pairs as “close” (within text line) and some others as “distant” (between lines), a distance metric can be automatically learned to fit the target of small distance within text line and large distance between lines.

Under a learned distance metric, the tree generated by MST algorithm has the characteristic that the neighboring components of the same line are connected and each line corresponds to a subtree (Fig. 1). However, the branches (paths between terminal and branching nodes) do not correspond to text lines perfectly due to the variability of layout of text lines. We hence use a second-stage clustering algorithm to dynamically cut the edges of the tree into groups corresponding to text lines.

![Fig. 1 MST of a document image.](image-url)

The criterion to select the edge to cut and the criterion to stop cutting (to determine the number of clusters) are important in the second stage. Simply deleting the shortest edge does not promise because the edges between different lines (red lines in Fig. 1) are not always longer than those within the same line. Our approach is to select the edge to cut such that the sum of hypervolumes of clusters is reduced most, and to stop clustering when the objective of text lines reaches a maximum [13].
From the description above, the framework of our approach can be depicted as in Fig. 2.

3.2. Clustering algorithm

Our algorithm starts with a binary document image. In pre-processing, the connected components are labeled using a recent fast algorithm based on contour tracing [14]. Small components with few black pixels are considered as noises and are removed. We then estimate the dominant character size from the component-size histogram obtained by the method in [8]. The components with height or width larger than three times of the dominant character height are split using the method in [15] because they are almost formed by touched multiple characters and such big components affect the result of MST clustering. Finally, each component is viewed as a node in a graph (document graph). Each pair of nodes is linked by an edge with the distance between them as the weight. The distance metric is designed to strengthen within-line links and weaken between-line links. From the weighted document graph, a minimal spanning tree (MST) is built using Kruskal’s algorithm [16]. In the resulting tree, most edges correspond to within-line links and some correspond to between-line links.

Since the Kruskal’s MST algorithm is well known and can be easily found in the literature, we will not give its details in this paper.

3.3. Text line grouping

Although the learned distance metric encourages the components in the same text line to be connected in a subtree, there are still some components from different lines connected. Such between-line edges are not obvious because their lengths (distances between components) are not necessarily longer than the within-line edge lengths. Fortunately, a criterion based on hypervolume can deal with this problem well [17]. We use the sum of hypervolumes of the clusters of connected components for evaluating the partition:

\[ F_v = \sum_{i=1}^{k} \left[ \det(C_i) \right]^{1/2}, \]

where \( \det(C_i) \) denotes the determinant of the covariance matrix \( C_i \) of cluster \( i \), which is computed from the constituent black pixels of the connected components in the cluster.

Initially, all the components in the MST are considered as a single cluster, and every edge is deleted tentatively to split the cluster into two clusters (each cluster corresponds to a disjoint subtree). The edge with the most reduction of \( F_v \) measure is selected and deleted such that the total \( F_v \) measure of the document is minimized [13]. We call this maximum hypervolume reduction criterion, denoted by:

\[ \text{edge}_{\text{deleted}} = \arg \max_{\text{edge}} \Delta F_v \]

\[ = \arg \max_{\text{edge}} [F_v(S_k) - F_v(S_{k+1})] \]

where \( S_k = \{T_1, T_2, \ldots, T_k\} \) denote the partition of \( k \) disjoint subtrees (\( S_1 \) denotes the initial MST).

The \( F_v \) measure cannot evaluate the number of clusters since it always decreases as the number of clusters increases. Fortunately, it is reasonable to assume rectangular shapes for the text lines (if a text line is curvilinear, it can be divided into several sublines that are approximately straight). We conjecture that when the number of clusters (partitioned text lines) is appropriate, a measure of the straightness of text lines reaches a maximum. We compute the total straightness measure as:

\[ S_{vm} = \sum_{i=1}^{k} \left( \frac{\lambda_{i1}}{\lambda_{i2}} \right)^2, \]

where \( k \) is the number of clusters, \( \lambda_{i1} \) and \( \lambda_{i2} \) (\( \lambda_{i1} \geq \lambda_{i2} \)) are the eigenvalues of the covariance matrix of each cluster.

Our previous experiments in [13] demonstrate that the number of clusters maximizing \( S_{vm} \) mostly fits the actual number of text lines.

4. Distance Metric Learning

As many clustering algorithms rely critically on a good metric between pairs of input units, the definition of the distance between connected components is the key to make the generated MST have the components of the same
line in a subtree and those of different lines in different subtrees. We were inspired by the work of distance metric learning of [18] and herein design our distance metric for text line separation by supervised learning.

4.1. Problem Formulation

For learning our distance metric between connected components, we need some training samples of component pairs labeled as “within line” and “between lines”. To do this, we annotated some training document images using our ground-truthing tool GTLC (Ground-truthing tool for Text Lines and Characters) [19], which label the text lines and characters by automated transcript alignment and hand correction.

Let \( C = \{x_1, x_2, \cdots, x_n\} \) be a collection of connected components in a training document, where \( n \) is the number of components. We obtain two sets of component pairs as the samples for metric learning:

\[
S = \{(x_i, x_j) \mid x_i \text{ and } x_j \text{ belong to the same line}\}, \quad D = \{(x_i, x_j) \mid x_i \text{ and } x_j \text{ belong to different lines}\}.
\]

Considering the fact that only the spatially neighboring components are linked in the minimal spanning tree, we can discard many component pairs from the sample set for accelerating metric learning. To do this, we construct the area Voronoi diagram [20] of the training document, which represents the spatial adjacency between the components. A component \( x_i \) is the neighbor of another one \( x_j \) only if they share a Voronoi edge on their boundaries. The pairs that are not adjacent in the Voronoi diagram are removed from \( S \) and \( D \).

The aim of metric learning is to make the distance between components in \( S \) small and the distance between components in \( D \) large under the learned metric. Hence, we formulate the problem of metric learning as a convex programming problem [21]:

\[
\min_{A \in R^{m \times m}} \sum_{(x_i, x_j) \in S} \|x_i - x_j\|_A^2
\]

\[\text{st} \quad A \succeq 0, \quad \sum_{(x_i, x_j) \in D} \|x_i - x_j\|_A^2 \geq 1,\]

where the matrix \( A \in R^{m \times m} \) (\( m \) is the dimensionality of the feature space characterizing component pairs) defines the distance metric:

\[
d(x_i, x_j) = d_A(x_i, x_j) = \|x_i - x_j\|_A = \sqrt{v_{ij}^T A v_{ij}},
\]

where \( v_{ij} \) is the feature vector characterizing the relation between points \( x_i \) and \( x_j \). \( A \) is determined by solving the convex programming problem.

4.2. Feature Space

The features characterizing the relation between two components \( x_i \) and \( x_j \), which are integral of the distance metric, are also influential to the performance of clustering. We use eight features listed below.

1) Normalized horizontal and vertical distances between the centroids of two components.

The horizontal/vertical distance between the centroids of two connected components measures the spatial closeness. For generalizing to different documents (with differing font size and imaging resolution), this distance should be normalized with respect to the character size (divided by the estimated dominant character size).

2) Normalized horizontal and vertical overlapping degree.

If two components overlap horizontally (align vertically), the normalized horizontal overlap degree can be computed by:

\[
\text{novlp}_x = \frac{1}{2} \left( \frac{\text{ovlp}_x}{W_1} + \frac{\text{ovlp}_x}{W_2} \right) - \frac{\text{dist}}{\text{span}}.
\]

where \( \text{ovlp}_x \) is the overlapping width of two bounding boxes, \( W_1 \) and \( W_2 \) are the widths of the bounding boxes, \( \text{dist} \) is the horizontal distance between the centers of two bounding boxes, and \( \text{span} \) is the spanning width of two bounding boxes (Fig. 3).

The normalized vertical overlap degree is computed similarly from the heights of two bounding boxes.

![Fig. 3 Definition of normalized horizontal overlap.](image)

3) Normalized horizontal and vertical minimum run-length.

The horizontal minimum run-length (MRL) is the horizontal run-length between vertically overlapping
(horizontally aligned) connected components, wherein the minimum horizontal distance between black runs is taken as the distance measure (Fig. 4). It is similarly normalized with respect to the dominant character size of the document image.

The vertical minimum run-length ($MRL_v$) is computed similarly and normalized with respect to the dominant character size.

4) Height and width ratio of merged components.
Suppose two connected components are merged, then the Height Ratio is computed by:

$$R_{hei} = \frac{\max(H_1, H_2)}{\text{span}},$$

where $H_1$ and $H_2$ are the heights of the bounding boxes, and $\text{span}$ is the spanning height of two bounding boxes.

The Width Ratio is computed similarly from the widths of two connected components.

5. Experimental Results

We evaluated the performance of our algorithms on a Chinese handwritten documents database HIT-MW [22], which was collected by Harbin Institute of Technology and is publicly available for free use. The database contains 853 text forms written by more than 780 writers. It has 8664 text lines and each line has 21.51 characters on average. Each document was scanned at a resolution of 300DPI.

Since the images in the HIT-MW database are not labeled at connected components level (only a part of images have been segmented into text lines), we have annotated 105 document images using our ground truthing tool GTLC [19]. We selected five documents from them for distance metric learning and used the remaining 100 documents containing 1106 text lines for testing.

We evaluate the performance by counting the number of match between the pixels of text lines detected by the algorithm and the pixels in the ground truth data. Similar to [6], we calculate the MatchScore between a detected text line and a ground truthed text line:

$$\text{MatchScore}(i, j) = \frac{T(G_i \cap R_j)}{T(G_i \cup R_j)},$$

where $G_i$ is the set of all pixels of the $i$-th ground truthed text line, $R_j$ is the set of all pixels of the $j$-th detected line, $T(S)$ is the cardinality of set $S$. The Hungarian algorithm [23] is used to find one-to-one correspondence between the detected text lines and the ground-truthed ones. The performance is evaluated at the text line level. If a ground-truthed line and the corresponding detected line share at least 90% of pixels, the detected text line is claimed to be correct.

The correct rates of text line detection by MST clustering with and without metric learning are given in Table 1. We can see that distance metric learning improves the performance of text line segmentation significantly.

<table>
<thead>
<tr>
<th>Table 1. Correct rates of text line detection.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Detected text lines</td>
</tr>
<tr>
<td>with learned metric</td>
</tr>
<tr>
<td>with metric by hand [13]</td>
</tr>
</tbody>
</table>

The algorithm with metric learning, though performs sufficiently well, still remains some text line detection failures. The failures are mostly of two types: (1) Error line splitting (ELS): a real text line is split into two or more lines (corresponding to multiple clusters); (2) Error line merging (ELM): two or more real text lines are merged into a single cluster.

We observed that the errors of line splitting (ELS) occurred when characters are inserted in a line, such as those (annotated by blue circles) in Fig. 5.

Fig. 5 An example of Error line splitting.

The errors of line merging (ELM) are mainly caused by the overlapping of two neighboring text lines, such as that (annotated by blue circle) in Fig. 6. In this case, since two lines are connected in only few touched characters, a post-processing procedure is necessary to separate them vertically.

Fig. 6 An example of Error line merging.

Fig. 7 An example of fully correct line extraction.

Fig. 7 shows an example of fully correct text line segmentation by the proposed method. Overall, the
proposed performs very well on handwritten documents with multi-skewed and curved text lines.

Our algorithm was implemented in C++ codes on a personal computer of Pentium 4-3.6 GHz. The overall processing speed, about 300 connected components (about 150 characters) per second, is nevertheless acceptable.

We cannot compare our results with the previous ones due to the unavailability of common dataset. Li et al. tested their level set based method on 100 handwritten Chinese documents with 1672 text lines of UMD [6]. They obtained a correct text line detection rate 92% and the computation speed on an image of 2000x1500 pixels is nearly 20 seconds on 1.6GHz CPU. Our correct detection rate, 95.02%, is competitively high, and the speed is much faster.

6. Conclusion

We propose a new method based on minimum spanning tree (MST) clustering with distance metric learning for handwritten text line segmentation. This bottom-up method can extract multi-skewed, curved and slightly overlapping text lines. The algorithm is free of artificial parameter, and supervised learning of distance metric improves the accuracy of text line detection significantly. Our experimental results on handwritten Chinese documents are competitive to the best ones reported in the literature.

Our algorithm is to be further improved by refining the features of distance metric and the post-processing procedure, and to be evaluated on a larger image database.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (NSFC) under grant no.60775004.

References