Proceedings of the 2007 International Conference on Wavelet Analysis and Pattern Recognition, Beijing, China, 2-4 Nov. 2007

LOCATING TEXT BASED ON CONNECTED COMPONENT AND SVM

JIN-LIANG YAO, YAN-QING WANG, LU-BIN WENG, YI-PING YANG

Institute of Automation, Chinese Academic of Sciences, Beijing 100084, P. R. China
E-MAIL: jinliang.yao@ia.ac.cn, yanqing.wang@ia.ac.cn, yiping.yang@ia.ac.cn

Abstract

This paper presents a novel connected component based method for locating text in complex background using support vector machine (SVM). Our method is composed of two stages. In the first stage, the cascade of threshold classifiers and support vector machine are used to identify characters. In the second stage, the identified characters are combined into texts, and then text features are extracted and used to identify text region. Two kinds of features which are character features and text features are utilized to locate text region. Character features are used to discriminate character connected components (CCs) from other objects in complex background. Text features describe the characteristics that characters in the same text have same size, color and font. The cascade of threshold classifiers can discard most non-character object, and improve the efficiency of character feature extraction. SVM is used to identify characters which the cascade of threshold classifiers can not identify. Experimental results demonstrate that the proposed approach is robust with respect to different character sizes, colors and languages, and achieves high precision which measured on the ICDAR 2003 test database.

Keywords: Text Location; support vector machine; cascade of classifiers; text features

1. Introduction

Texts in images carry useful and important information in daily life, such as road signs, and advertisement, etc, so a robust text reading system derives many applications. For instance, this system can be used in content-based images or video sequences retrieval. Also this system can help a visually handicapped person to increase environmental awareness. However, current commercial OCR (Optical Character Recognition) systems do not work well on texts embedded in images with complex background because they cannot locate text regions in images. Automatic text location is very important to recognize text in images with complex background.

In recent years, there have been many methods on detecting and locating text in images. These methods can be broadly categorized into two types: connected-component-based (region-based) and texture-based methods. The connected-component-based method assumes that pixels of each character have similar color and can be segmented from the background by color segmentation. Connected component (CC) analysis is applied to obtain candidate character CCs on the result of segmentation. Various features of candidate character CCs such as size and aspect ratio are extracted and used to verify whether CCs are characters or not. J. Ohya [1] uses adaptive thresholding to segment an image. Additional heuristics are applied to verify a character or a part of a character. Hinnerk Becker [7] uses an adaptive binarization method to extract character regions which are then combined to lines of text following some geometrical constraints. A number of (mostly edge or histogram based) features is calculated to classify these hypothesis as text or non-text. The strength of CC-based methods is more effective for characters of large size than texture based methods. The texture-based method assumes that text in images is a special texture. Many texture features in blocks of image are used to classify blocks into text or non-text. Complicated classification methods such as neural network and SVM are often utilized in these methods. K I Kim [2] uses SVM to recognize character pixels, and then continuously adaptive mean shift algorithm is used to obtain text regions. Alex Chen [8] extracts texture features of block, and then uses Adaboost classifier to recognize text block. The strength of these methods is effective for small characters. The weakness of these methods is that they are difficult to obtain the refined bounding box of text region.

In this paper, we present a new CC based method for locating text in images with complex background. Block diagram is shown in Fig. 1. In the stage of image decomposition, an improved locally adaptive thresholding method is proposed in which a threshold is calculated over each pixel by a local mean. The proposed thresholding method decomposes an image into three layers which are white, gray, and black layer. Then connected component analysis is used on the white and black layer respectively to get candidate character CCs. In the stage of character identification, character features are extracted from CCs, and used to discard most non-character CCs by a cascade of threshold classifiers. CCs which are not discarded classifiers cascade are fed into a SVM to classify into character CCs or not. In the last stage, the character CCs verified by SVM will be merged into candidate text regions on the criterion that the CCs are neighbor each other and have approximate attributes such as stroke width, height. Text features are extracted to verify candidate text regions.

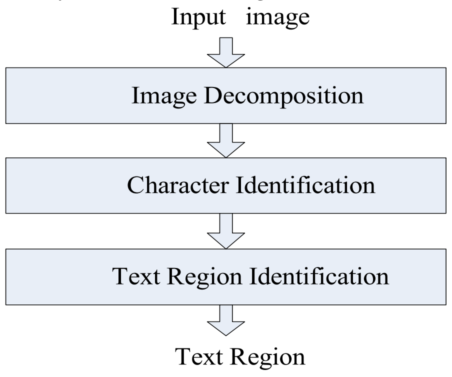


Fig. 1. The block diagram of our algorith

The rest of the paper is organized as follows: In section 2, we introduce our locally adaptive thresholding method. Section 3 describes how to extract various character features from CCs and effectively identify character CCs using SVM and the cascade of threshold classifiers. Section 4 gives how to obtain text region from character CCs and verify ones. In section 5, the performance analysis and the experimental results are provided. Finally, the conclusion is given in section 6.

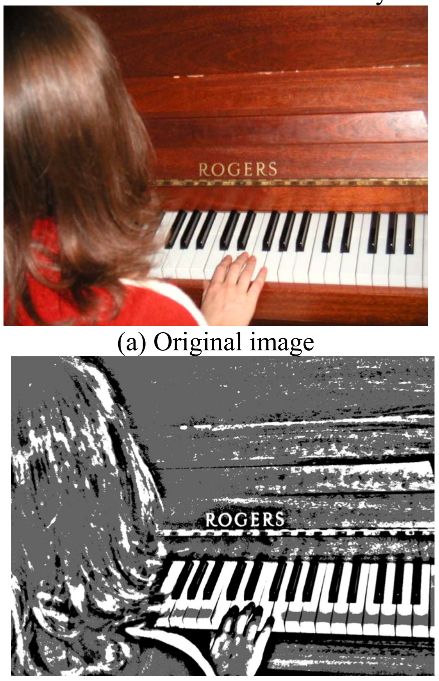
2. Image Decomposition

The goal of image decomposition is to cluster all pixels belong to a character into a CC. First, image segmentation is used. Then CCs are obtained by CC analysis or CC labeling on the result of segmentation. There are many image segmentation methods used in text locating such as color clustering. These methods are effective but high computational. Since characters in images usually have sharp edge and similar color, we propose a locally adaptive threshold method which a threshold is calculated over each pixel by a local mean. This algorithm can be described by the equation (1):

$$\text{Segment Result}(x,y)=\begin{cases}255 & I(x,y)>T_{+}(x,y)\\ 0 & I(x,y)<T_{-}(x,y)\\ 100 & other\end{cases} \qquad (1)$$

$$T_{\pm}(x,y)=Mean(x,y,W_B)\pm offset$$

Where I(x,y) is a pixel in input gray image. $T_{\pm}(x,y)$ are the local thresholds corresponding to I(x,y). $Mean(x,y,W_B)$ is the mean of the pixels in the block whose center is I(x,y), and the size of block is W_B. The "offset" is a positive integer by which many pixels are segmented into the gray layer. The proposed segmentation method is faster than Niblack method which requires calculating a locally variance. In our experiment, the k, "offset" and W_B are respectively set to be 0.2, 10 and 71 based on experiments. 255, 100 and 0 respectively correspond to three layers: white, gray and black layer. Fig. 2 shows the result for our proposed segmentation method. Since pixels belong to characters have high locally variance, "offset" can be considered as a threshold of variance which eliminates non-character pixels which have low variance. So the pixels in gray layer do not belong to character pixels. The proposed segmentation method only needs to calculate a locally mean and can eliminate many non-character pixels, so it improves the efficiency of our system. The pixels in white or black layer are candidate character pixels. CC analysis are utilized to obtain candidate CCs in the white and black layers.



(a) Original image

(b) Result image

Fig. 2 The result of improved locally threshold segmentation

3. Character Identification

In previous work, SVM is often used to locate text in texture-based methods. In our method, SVM is applied to identify character CCs. SVM is a trainable classifier which can achieve high classification accuracy that is proven in other classification application such as face detection. However, SVM requires more processing time than other classifiers such as neural network in the test stage. And extracting character features of all CCs needs more computational time. To meet real-time requirement for text locating, a cascade of threshold classifiers is used in the feature extraction stage to discard most obvious non-character CCs.

3.1. SVM

In this section we briefly sketch the SVM algorithm. A more detailed description of SVM can be found in [3]. SVM is based on the principle of structural risk minimization which different from the idea of minimizing the training error. From the implementation point of view, training a SVM is equivalent to solving a linearly constrained quadratic programming problem in a number of variables equal to the number of data points. Given a training set (x_i, y_i), where $x_i \in R^n$ is input pattern and $y_i \in \{1,-1\}$ is the corresponding desired label, i = 1,…,L. SVM requires the solution of the following optimization problem:

$$\bar{\alpha} = \arg\min_{\alpha} \frac{1}{2}\sum_{i=1}^{L} \alpha_i \alpha_j y_i y_j K(x_i, y_j) - \sum_{i=1}^{L} \alpha_i \qquad (2)$$

Subject to $\sum_{i=1}^{L} \alpha_i y_i = 0, 0 \le \alpha_i \le C$

Where C > 0 is the penalty parameter of the error term, $K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$ is called the kernel function. α_i is a Lagrange multiplier. Training vector x_i is mapped into a higher dimensional space by the function $\varphi(x_i)$. SVM finds a linear separating hyper-plane with the maximal margin in this higher dimensional space. All the x_i corresponding to non-zero $\bar{\alpha}$ are called support vectors (SVs). The classifier function is as follow:

$$f(x) = \mathrm{sign}\left(\sum_{x_i \in SVs} \bar{\alpha}_i y_i K(x_i, x) + \bar{b}\right) \qquad (3)$$

Where $\bar{b} = -\frac{1}{2}\sum_{x_i \in SVs} \bar{\alpha}_i y_i [K(x_r, x_i) + K(x_s, x_i)]$, x_r and x_s are different types SVs.

3.2. Character Feature Extraction

Three types of character features are extracted from CCs. They are basic character features, contour roughness feature and stroke width features. These features describe the characteristic of character CCs that are different from non-character CCs.

Basic character features are easily obtained which are area, height, width, and perimeter. Furthermore, four new features are constructed by equation (4) based on experience.

$$\begin{aligned} \mathrm{Occupy\_Rate} &= \frac{\mathrm{area}}{(\mathrm{height} * \mathrm{width})} \\ \mathrm{Aspect\_Ratio} &= \frac{\max(\mathrm{width}, \mathrm{height})}{\min(\mathrm{width}, \mathrm{height})} \\ \mathrm{Compactness} &= \frac{\mathrm{area}}{(\mathrm{perimeter} * \mathrm{perimeter})} \end{aligned} \qquad (4)$$

The features of area and perimeter are very effective to discard small non-character CCs. Aspect_Ratio feature can be utilized to remove too thin CCs. Compactness feature describes the characteristic that characters are composed of stripes. The Compactness of character CCs are smaller than non-character CCs. Compactness feature can effectively discard non-character CCs which are not composed of stripes.

Contour roughness features are important to discard those non-character CCs which contours are irregular compared with character contours. 8 neighborhood pixels of the pixel on the contour are used to examine whether the pixel is a rough one or not. The centre pixels of fig. 3 are considered as rough pixels. "1" denotes a pixel in the CC, "0" is a background pixel.

1	0	0
0	1	0
0	0	0

1	0	0
1	1	0
1	0	0

Fig.3 Two examples of the rough pixel templates

180 templates of rough pixel such as Fig.3 are defined in our algorithm. If the contour pixel of CCs matches a rough pixel template, it is considered as a rough pixel. According to the number of the rough pixels of the CCs (Num_RoughPixels), the feature of Contour_Roughness is defined as follows:

$$Contour\_Roughness = \frac{Num\_RoughPixels}{height + width} \qquad (5)$$

We all know that characters are composed of strokes which have consistent stroke width. The stroke width is the distinct feature of character CCs which are different from other object in background. A fast parallel thinning algorithm [4] is performed to obtain the median axis of CC. At the same time, the count of iteration to verify the pixel on the median axis in the algorithm can be considered as the stroke width of pixel. The average of stroke width of pixels on the median axis is called as stroke width of CC, and the variance is the stroke width variance of CC. The maximum of stroke width as well as the number of pixels at the median axis are also obtained. In order to reduce the influence of different character sizes, we combine stroke width features and the size of CCs to obtain four scale invariability features, as follows:

$$\begin{aligned} \mathrm{StrokeWidth\_Size\_Ratio} &= \frac{\mathrm{Stroke\ Width}}{\mathrm{Max(height, width)}} \\ \mathrm{MaxStrokeWidth\_Size\_Ratio} &= \frac{\mathrm{maximum\ of\ stroke\ width}}{\mathrm{Max(height, width)}} \\ \mathrm{StrokeWidthVariance\_Ratio} &= \frac{\mathrm{Variance\ of\ Stroke\ Width}}{\mathrm{Stroke\ width}} \\ \mathrm{Stroke\_Density} &= \frac{\mathrm{Number\ of\ the\ median\ axis}}{\mathrm{height + width}} \end{aligned} \qquad (6)$$

In order to reduce the computational time for obtaining all character features, a cascade of threshold classifiers is used. Each of these features corresponds to a simple threshold classifier. When a new feature of CC is obtained, the corresponding threshold of the feature is utilized to identify whether the CC is character CC or not. If the CC is not a character CC, it is discarded, and is not required to calculate other features. The cascade of classifiers can obviously improve the efficiency of feature extraction and reduce the computational burden of SVM by discarding distinct non-character CCs.

The thresholds of classifiers are crucial to identify character CCs. In our approach, the cascade of threshold classifiers is used to remove those distinct non-character CCs and improve the speed of text locating. So we consider the maximum and minimum of every feature of character CCs in the train image database as the low and upper threshold of corresponding feature. These thresholds which are relax constraint can ensure that all character CCs can be recalled in the train database and have high recall rate on the test database. The ICDAR 2003 train database is used as our train database. The result of the cascade of threshold classifiers is shown in Fig. 4(a).

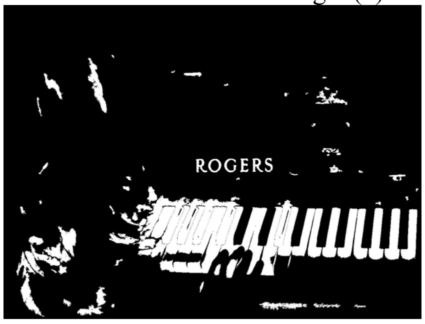
3.3. SVM based Characters Identification

SVM is trained on a database consisting of 4374 character and 4374 non-character samples, using the software package LIBSVM [5]. Each feature vector is composed of 12 character features which are got by above features extraction method. The feature vector firstly is linearly scaled to the range [0, 1]. Radial basis function is used as kernel function of SVM.

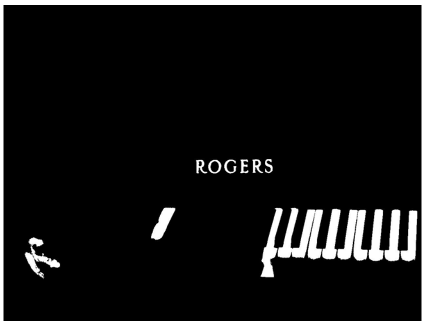
$$K(x, x_j) = \exp\{-\frac{|x - x_j|}{2\sigma^2}\}$$

Where the kernel bandwidth σ is determined through cross-validation, the penalty parameter of the error C is set to 2000. Character CC is defined to be +1, while non-character CC -1. The trained SVM has 1512 SVs among which the number of positive SVs is 397.

The remaining CCs through the cascade of threshold classifiers have character characteristic on a single feature. All character features of CCs are fed into the trained SVM to classify these CCs into character CCs or not. If the output of a feature vector is positive, it is classified as character CC. if the output is negative, it is not non-character CC. The result of SVM classification is shown in Fig. 4(b).



(a)Result of classifiers cascade



(b) Result of SVM classification

Fig.4. Result of character identification

4. Text Region Identification

The features of text describe the characteristic of all characters in the same text region. Before extracting text region features, we merge the neighboring character CCs into text regions with two constraints: position and attribute constraint. The position constraint defines the position relation of two CCs which may be merged into a text region. The attribute constraint means that the attributes of the CCs such as stroke width should be consistent if two CCs can be merged into a same text region. If some CCs are not merged into candidate regions, much stricter thresholds of the CC features are used to classify whether it is character CC or not. The detail conditions of merging into a text region are as follows. CC_i and CC_j define different CCs in the candidate CCs set. CC_j_XXX defines the attribute of the CC_i, for example, CC_j_Width defines the width of CC_j.

A. Position constraint:

$$\begin{aligned} &\text{MinWidth} = \text{Min}(CC_i\_\text{Width}, CC_j\_\text{Width}) \\ &(CC_i\_\text{Bottom} - CC_j\_\text{top}) > k_1 * \text{MinWidth} \qquad (7) \\ &(CC_j\_\text{Bottom} - CC_i\_\text{top}) > k_1 * \text{MinWidth} \end{aligned}$$

Equation (7) ensures these two CCs are in the horizontal line, the k_1 is a parameter which control the slope of the line.

$$\begin{aligned} &CC_i\_\text{Right} - CC_j\_\text{Left} > k_2 * \text{MinWidth} \qquad (8) \\ &\| \; CC_j\_\text{Right} - CC_i\_\text{Left} > k_2 * \text{MinWidth} \end{aligned}$$

Equation (8) verifies these two CCs are neighbor. k_2 controls the distance of the two CCs.

B. Attribute constraints:

$$\begin{aligned} &CC_i\_\text{GreyValue} - CC_j\_\text{GreyValue} < k_3 \\ &C_i\_\text{StrokeWidth} - CC_j\_\text{StrokeWidth} < k_4 \\ &k_5 * \text{MinWidth} > \text{MaxWidth} \qquad (9) \\ &k_5 * \text{MinHeight} > \text{MaxHeight} \end{aligned}$$

If two CCs meet all upper constraints (7)(8)(9), they can be merged into a same text region. All couples in the

CCs set are identified whether they are in the same text region. Then all CCs that are in the same text region are merged into a text region.

Five features of text region are extracted in our method. They describe the texture characteristics of text region. Four features are the variance of stroke width, gray value, width and height which can be obtained by calculating the variances of the corresponding character features of CCs in the candidate text region. If the candidate text region is a text region, the four variances are small. Furthermore, the ratio of average width and height of CCs which is in the same text region is regarded as a feature of text region. We use five experimental thresholds to examine whether a candidate text region is a text region or not.

5. Experiment Results

To evaluate the performance of the proposed method, we use the database which is available with the occasion of the ICDAR 2003 Robust Reading Competition [6]. This dataset contains a total of 508 realistic images with complex background. All Images in the dataset were captured with a variety of digital cameras. Cameras were used with a range of resolution and other settings, with the particular settings chosen at the discretion of the photographer. The texts in pictures are scene text. The images are organized in three sections: Sample, Trial and Competition. Only the first two are publicly available, the third set of images being kept separate by the competition organizers to have a completely objective evaluation. The Trial directory has two subdirectories: Trial-Train and Trial-Test. The Trial-Train images are used to train and tune our algorithm. The images in Trial-Test are used to evaluate our algorithm.

We also use the evaluation system which ICDAR text locating competition organizers propose. The evaluation system is based on the notions of precision and recall. For text locating it is unrealistic to expect a system to agree exactly with the bounding rectangle for a word identified by a human tagger. For each rectangle in the set of estimates, the closest match in the set of targets m(r_e,T) is found, and vice versa. The precision (p) and recall (r) are defined as follow:

$$p=\frac{\sum_{r_e\in E} m(r_e,T)}{|E|}\ ,\qquad r=\frac{\sum_{r_t\in T} m(r_t,E)}{|T|}$$

Where E is the set of estimates of algorithm, T is the set of ground-truth rectangles in the test image.

In ICDAR 2005 text locating competition [7], The two leading systems are submitted by Hinnerk Becker and Alex Chen respectively. The two systems are evaluated in the private test database. Our system is tested on the Trial-Test dataset. The column labeled t(s) gives the average time in seconds to process each image for each system. Hinnerk Becker and Alex Chen systems were made using a 2.4 ghz, However, our method using a 1.7 ghz PC. The performance of our method is shown in Table 1.

Table 1 Overall performance

System	P	R	f	t(s)
Hinnerk Becker	0.62	0.67	0.62	14.4
Alex Chen	0.60	0.60	0.58	0.35
Our Method	0.64	0.60	0.61	1.1

Although our system and the two leading system in 2005 text locating competition use different test datasets, the result of our method also can show the strength of our method due to the two database are very similar. The proposed method achieves a higher precision. However the recall is lower than the systems of Hinnerk Becker. Processing time is 13 times faster than the system of Hinnerk Becker. From the experiment, we found that image decomposition largely affects the effectiveness of our method. In our method, there are thirty images which get zero recall, due to the texts in the images do not meet the hypothesis that pixels of each character have similar color. Fig.5 shows two results of text detection. The black rectangles are the detected results.

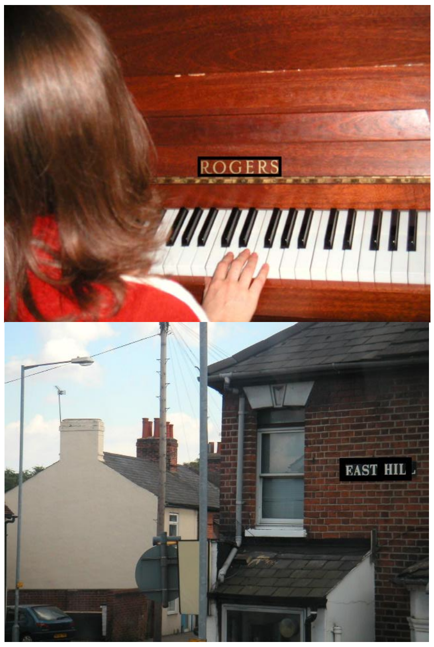


Fig.5 Final results of two images in test dataset

6. Conclusions

In this paper, we present a new method to locate text in images with complex background. In the proposed method, a locally adaptive thresholding method based on a locally mean and a offset is proposed. The proposed thresholding method is very efficient to text segmentation and can remove a lot of non-character pixels directly. A cascade of threshold classifiers and SVM are used to classify character CCs. The cascade of threshold classifiers can remove many distinct non-character CCs and improve the efficiency of our system, and SVM can identify character CCs which threshold classifiers can not classify. Furthermore, we propose new text features to classify text regions. Future work will focus on new image segmentation methods for text.

Acknowledgements

This research is supported in part by Research and Development of Modern Services Infrastructure Services (No.2006BAH02A01) under Chinese National Key Technology R&D Program.

References

[1] J. Ohya, A. Shio and S. Akamatsu, "Recognizing Characters in Scene Images", IEEE Trans. On PAMI, Vol. 16, No.2, pp.214-224, 1994

[2] K. I. Kim, K. Jung, H. J. Kim, "Texture-based approach for text detection in images using support vector machines and continuously adaptive mean shift algorithm", IEEE Trans. On PAMI, Vol.25, No.12, pp.1631-1639, 2003

[3] Burges CJC, "A Tutorial on Support Vector Machines for Pattern Recognition", Knowledge Discovery and Data Mining，Vol.2, No.2, pp.121-167, 1998

[4] T. Y. Zhang, C. Y. Suen, "A fast parallel algorithm for thinning digital patterns", Communication of the ACM, Vol.27, No.3, pp.236-239, 1984

[5] Chih-Chung Chang and Chih-Jen Lin, LIBSVM : a library for support vector machines, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm

[6] Simon M. Lucas, et al. "ICDAR 2003 robust reading competitions: entries, results, and future directions", IJDAR, Vol.7, No.2, pp. 105-122, 2005

[7] Simon M. Lucas, "ICDAR 2005 Text Locating Competition Results", In Proc. of the Eighth International Conference on Document Analysis and Recognition, Korea, pp. 80-84, 2005.

[8] X. Chen, A. L. Yuille, "Detecting and reading text in natural scenes", In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages II:366 –II:373, 2004.