Robust Text Detection in Natural Scene Images

Xu-Cheng Yin, *Member, IEEE,* Xuwang Yin, Kaizhu Huang, and Hong-Wei Hao

Abstract—Text detection in natural scene images is an important prerequisite for many content-based image analysis tasks. In this paper, we propose an accurate and robust method for detecting texts in natural scene images. A fast and effective pruning algorithm is designed to extract Maximally Stable Extremal Regions (MSERs) as character candidates using the strategy of minimizing regularized variations. Character candidates are grouped into text candidates by the single-link clustering algorithm, where distance weights and clustering threshold are learned automatically by a novel self-training distance metric learning algorithm. The posterior probabilities of text candidates corresponding to non-text are estimated with a character classifier; text candidates with high non-text probabilities are eliminated and texts are identified with a text classifier. The proposed system is evaluated on the ICDAR 2011 Robust Reading Competition database; the f-measure is over 76%, much better than the state-of-the-art performance of 71%. Experiments on multilingual, street view, multi-orientation and even born-digital databases also demonstrate the effectiveness of the proposed method. Finally, an online demo of our proposed scene text detection system has been set up at http://prir.ustb.edu.cn/TexStar/scene-text-detection/.

Index Terms—Scene text detection, maximally stable extremal regions, single-link clustering, distance metric learning

1 Introduction

Text in images contains valuable information and is exploited in many content-based image and video applications, such as content-based web image search, video information retrieval, and mobile based text analysis and recognition [1]–[5]. Due to complex background, and variations of font, size, color and orientation, text in natural scene images has to be robustly detected before being recognized and retrieved.

Existing methods for scene text detection can roughly be categorized into three groups: sliding window based methods [6]–[8], connected component based methods [9]–[13], and hybrid methods [14]. Sliding window based methods, also known as region-based methods, use a sliding window to search for possible texts in the image and then use machine learning techniques to identify text. These methods are slow as the image has to be processed in multiple scales. Connected component based methods extract character candidates from images by connected component analysis followed by grouping character candidates into text; additional checks may be performed to remove false positives. The hybrid method presented by Pan *et al*. [14] exploits a region detector to detect text candidates and extracts connected components as character candidates by local binarization; non-characters are eliminated with a Conditional Random Fields model [15], and characters can finally be grouped into text. More recently, Maximally Stable Extremal Regions (MSERs) based methods, which can be categorized as connected component based methods but using MSERs [16] as character candidates, have become the focus of several recent works [17]–[23].

MSER-based methods have reported promising performance on the widely used ICDAR 2011 Robust Reading Competition database [17]. However, several problems remain to be addressed. First, the MSERs algorithm detects a large number of repeating components and these repeating components are problematic for the latter character grouping algorithm. Most of the repeating components, apart from the components that most likely correspond to characters, need to be removed before further processing. The existing methods for MSERs pruning [19]–[21] still have room for improvements in terms of accuracy and speed. Second, current approaches [14], [19], [20] for text candidates construction, which can be categorized as rule-based and clustering-based methods, work well but are still not sufficient: rule-based methods generally require tuning parameters by hand, which is time-consuming and error-prune; the clustering-based method [14] shows good performance but is complicated by incorporating post-processing stage after minimum spanning tree clustering.

In this paper, we propose a robust and accurate MSER-based scene text detection method. First, by exploring the hierarchical structure of MSERs and adopting simple features, we design a fast and accurate MSERs pruning algorithm; the number of character candidates to be processed is significantly reduced with a high accuracy.

- *X.-C. Yin is with the Department of Computer Science and Technology and also with the Beijing Key Laboratory of Materials Science Knowledge Engineering, School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing 100083, China. E-mail: xuchengyin@ustb.edu.cn.*
- *X. Yin is with the Department of Computer Science and Technology, School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing 100083, China. E-mail: xuwangyin@gmail.com.*
- *K. Huang is with the Department of Electrical and Electronic Engineering, Xi'an Jiaotong-Liverpool University, Suzhou 215123, China. E-mail: kaizhu.huang@xjtlu.edu.cn.*
- *H.-W. Hao is with the Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China. E-mail: hongwei.hao@ia.ac.cn.*

Manuscript received 19 Jan. 2013; revised 20 Aug. 2013; accepted 15 Sep. 2013. Date of publication 26 Sep. 2013. Date of current version 29 Apr. 2014. Recommended for acceptance by R. Manmatha.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier 10.1109/TPAMI.2013.182

0162-8828 © 2013 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

Second, we propose a novel self-training distance metric learning algorithm that can learn distance weights and clustering threshold automatically; character candidates are clustered into text candidates by the single-link clustering algorithm using the learned parameters. Third, we propose to use a character classifier to estimate the posterior probabilities of text candidates corresponding to non-text and remove text candidates with high non-text probabilities. Such elimination helps to train a more powerful text classifier for identifying text. Finally, by integrating the above ideas, we build an accurate and robust scene text detection system. The system is evaluated on the benchmark ICDAR 2011 Robust Reading Competition database (Challenge 2) and has achieved an f-measure of 76%, which is much higher than the current best performance of 71%. Moreover, experiments on multilingual, street view, multi-orientation and even born-digital (web and email) databases also demonstrate that our method achieves considerable improvements over existing methods. An online demo of our proposed scene text detection system is available at *http://prir.ustb.edu.cn/TexStar/scene-text-detection/*.

The rest of this paper is organized as follows. Recent MSER-based scene text detection methods are reviewed in Section 2. Section 3 describes the proposed scene text detection method. Section 4 presents experiments on components (MSERs pruning, distance metric learning and text candidates elimination) of the proposed method and the overall system performance on several public databases. Final remarks are presented in Section 5.

2 Related Work

As described above, MSER-based methods have demonstrated very promising performance in many real projects. However, current MSER-based methods still have some key limitations, i.e., they may suffer from detecting of repeating components and also insufficient text candidates construction algorithms. In this section, we will review the MSER-based methods focusing on these two problems. Other scene text detection methods can be referred to some survey papers [24]–[26].

The main advantage of MSER-based methods over traditional connected component based methods roots in the usage of the MSERs algorithm for character extraction – the MSERs algorithm is able to detect most characters even when the image is in low quality (low resolution, strong noises, low contrast, etc.). However, one severe but not so obvious pitfall of the MSERs algorithm is that most of the detected MSERs are in fact repeating with each other. Repeating MSERs are problematic for the latter character candidates grouping algorithm, thus most of the repeating MSERs, apart from the MSERs that most likely correspond to character, need to be removed before being fed to the character grouping algorithm. The MSERs pruning problem has been studied by Carlos *et al*. [19] and Neumann and Matas [20], [21]. Carlos *et al*. [19] presented a MSERs pruning algorithm that contains two steps: (1) reduction of linear segments by maximizing the *border energy* function; and (2) hierarchical filtering with a cascade of filters. Neumann and Matas [21] proposed a MSER++ based text detection method, which exploits rather complicated features, e.g., higher-order properties of text and uses exhaustive search for pruning. Later, Neumann and Matas [20] presented a two-stage algorithm for Extremal Regions (ERs) pruning with the exhaustive search strategy. In the first stage, a classifier trained from incrementally computable descriptors (*area, bounding box, perimeter, Euler number and horizontal crossing*) is used to estimate the class-conditional probabilities of ERs; ERs corresponding to local maximum of probabilities in the ER inclusion relation are selected. In the second stage, ERs passed the first stage are classified as characters and non-characters using more complex features. The above methods all explore the hierarchical structure of MSERs, but have used different methods for estimating the probabilities of MSERs corresponding to characters. To deal with the large number of repeating MSERs, they have used relevant features (cascading filters and incrementally computable descriptors) in pruning.

Another problem with MSER-based methods, or more generally, connected component based methods and hybrid methods, is the absence of an effective text candidates construction algorithm. The existing methods for text candidates construction fall into two general approaches: rule-based [18]–[20] and clustering-based methods [14]. Neumann and Matas [20] grouped character candidates by meanings of the text line constrain. Their basic assumption is that characters in a word can be fitted by one or more top and bottom lines. The text line constrain is quite elaborate, but it's too restrictive for polluted text, handwritten text and other language model. Carlos *et al*. [19] constructed a fully connected graph over character candidates, filtered edges by running a set of tests (edge angle, relative position and size difference of adjacent character candidates) and used the remaining connected subgraphs as text candidates. Chen *et al*. [18] pairwised character candidates as clusters with constrains on stroke width and height difference, and exploited a straight line to fit the centroids of clusters. They declared a line as text candidate if it connected three or more character candidates. The clustering-based method presented by Pan *et al*. [14] clusters character candidates into a tree using the minimum spanning tree algorithm with a learned distance metric [27]; text candidates are constructed by cutting off between-text edges with an energy minimization model. The above rule-based methods generally require hand-tuned parameters, while the clustering-based method is complicated by the incorporating of the post-processing stage, where one has to specify a rather complicated energy model.

3 Robust Scene Text Detection

By incorporating several key improvements over traditional MSER-based methods, we propose a novel MSER-based scene text detection method. The structure of the proposed system, as well as the sample result of each stage is presented in Fig. 1. The proposed scene text detection method includes the following stages:

1) *Character candidates extraction*. Character candidates are extracted using the MSERs algorithm; most of the repeating components are removed by the proposed MSERs pruning algorithm by minimizing regularized variations. More details are presented in Section 3.1.

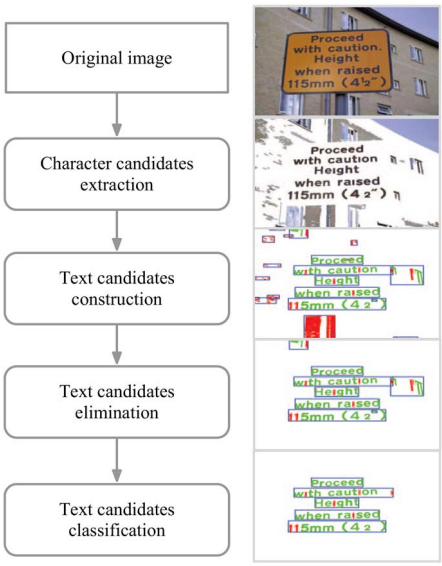


Fig. 1. Flowchart of the proposed system and results after each step of the sample. Text candidates are labeled by blue bounding rectangles; character candidates identified as characters are colored green, and others red.

2) *Text candidates construction*. Distance weights and clustering threshold are learned simultaneously using the proposed metric learning algorithm; character candidates are clustered into text candidates by the single-link clustering algorithm using the learned parameters. More details are presented in Section 3.2.

3) *Text candidates elimination*. The posterior probabilities of text candidates corresponding to non-texts are estimated using the character classifier and text candidates with high non-text probabilities are removed. More details are presented in Section 3.3.

4) *Text candidates classification*. Text candidates corresponding to true texts are identified by the text classifier. An AdaBoost classifier is trained to decide whether an text candidate corresponding to the true text or not [28].

3.1 Character Candidates Extraction

3.1.1 Pruning Algorithm Overview

Repeating components is the major pitfall when the MSER algorithm is applied as a character segmentation algorithm. Considering the MSERs tree presented in Fig. 3(a), this figure shows that fourteen MSERs are detected for the word "PACT" but only four of them are really interested to us. The hierarchical structure of MSERs is quite useful for designing a pruning algorithm. As characters cannot "contain" or be "contained" by other characters in real world, it is safe to remove children once the parent is known to be a character, and vice versa. If the MSERs tree is pruned by applying this kind of parent-children elimination operation recursively, we are still "safe" and all characters are preserved after the elimination. As an example, Fig. 3(e) shows that a set of disconnected nodes containing all the desired characters can be extracted by applying this algorithm to the MSERs tree in Fig. 3. However, it can be computationally expensive to identify characters, which usually entails the computations of complex features.

Fig. 2. Character correspondence in MSERs trees. (a) A MSERs tree whose children corresponds to characters. (b) A MSERs tree whose parent corresponds to character.

Fortunately, rather than identifying the character, we can simply choose the one that is more likely to be characters in a parent-children relationship. [21] claimed that such pairwise relationships may not be sufficient to eliminate non-character MSERs, and pruning should exploit some complicated higher-order properties of text. Alternatively, our empirical study indicates that this probability can be fast estimated using our regularized variation scheme with reasonable accuracy. As there are different situations (one-child and multiple-children) in MSERs trees, we design two algorithms based on the parent-children elimination operation, namely the *linear reduction* and *tree accumulation* algorithm. The linear reduction algorithm is used to remove line segments in the MSERs tree at first and the accumulation algorithm is then used to further remove repeated characters.

3.1.2 Variation and Its Regularization

According to Matas *et al*. [16], an "extremal region" is a connected component of an image whose pixels have either higher or lower intensity than its outer boundary pixels. Extremal regions of the whole image are extracted as a rooted tree. An extremal region is in fact a set of pixels. Its variation is defined as follows. Let R_l be an extremal region, $B(R_l) = (R_l, R_{l+1}, \ldots, R_{l+\Delta})$ (Δ is a parameter) be the branch of the tree rooted at R_l, the variation (instability) of R_l is defined as

$$\mathcal{V}(R_l) = \frac{|R_{l+\Delta} - R_l|}{|R_l|}, \tag{1}$$

where $|R|$ denotes the number of pixels in R. An extremal region R_l is a maximally stable extremal region if its variation is lower and more stable than its parent R_{l-1} and child R_{l+1} [16]. Informally, a maximally stable extremal region is an extremal region whose size remains virtually unchanged over a range of intensity levels [20].

As MSERs with lower variations have sharper borders and are more likely to be characters, one possible strategy for the parent-children elimination operation is to select parent or children based on who have the lowest variation. However, this strategy alone will not work because MSERs corresponding to characters may not necessarily have lowest variations. Consider a very common situation depicted in Fig. 2. The children of the MSERs tree in Fig. 2(a) correspond to characters while the parent of the MSRE tree in Fig. 2(b) corresponds to a character. The "minimizing variation" strategy cannot deal with this situation because either the parent or children may have the lowest variations. However, our experiment shows that this difficulty can be easily overcome by variation regularization. The basic idea is to penalize variations of MSERs with too large or too small aspect ratios. Note that we are not requiring characters to have the lowest variations globally, and

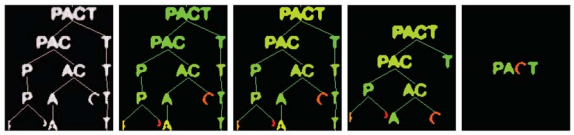


Fig. 3. The process of MSERs pruning. (a) MSERs tree of a text segment. (b) MSERs colored according to variations, as variations increase, MSERs are colored from green to yellow then to red. (c) MSERs colored according to regularized variations. (d) MSERs tree after linear reduction. (e) Character candidates after tree accumulation.

a lower variation in a parent-children relationship suffices for our algorithm.

Let $\mathcal{V}$ be the variation and a be the aspect ratio of a MSER, the aspect ratios of characters are expected to fall in $[a_{min}, a_{max}]$, the regularized variation is defined as

$$\mathcal{V} = \begin{cases} \mathcal{V} + \theta_1(a - a_{max}) & \text{if } a > a_{max} \\ \mathcal{V} + \theta_2(a_{min} - a) & \text{if } a < a_{min} \\ \mathcal{V} & \text{otherwise} \end{cases}, \tag{2}$$

where θ_1 and θ_2 are penalty parameters. Based on experiments on the training database, these parameters are set as $\theta_1 = 0.01, \theta_2 = 0.35, a_{max} = 1.2$ and $a_{min} = 0.3$.

Fig. 3(b) shows a MSERs tree colored according to variation. As variation increases, the color changes from green to yellow then to red. The same tree colored according to regularized variation is shown in Fig. 3(c). The MSERs tree in Fig. 3(c) are used in our linear reduction (see Fig. 3(d)) and tree accumulation algorithm (see Fig. 3(e)). Notice that "variation" in the following refers to the variation after regularization.

3.1.3 Linear Reduction

The linear reduction algorithm is used in situations where MSERs has only one child. The algorithm chooses from parent and child the one with the minimum variation and discards the other. This procedure is applied across the whole tree recursively. The detailed algorithm is presented in Fig. 4. In short, given a MSERs tree, the procedure returns the root of the processed tree whose linear segments are reduced. The procedure works as follows. Given a node t, the procedure checks the number of children of t; if t has no children, return t immediately; if t has only one child, get the root c of child tree by first applying the linear reduction procedure to the child tree; if t has a lower variation compared to c, link c's children to t and return t; otherwise return c; if t has more than one children, process these children using linear reduction and link the resulting trees to t before returning t. Fig. 3(d) shows the resulting MSERs tree after applying linear reduction to the tree shown in Fig. 3(c). Note that in the resulting tree all linear segments are reduced and non-leaf nodes always have more than one children.

3.1.4 Tree Accumulation

The tree accumulation algorithm is used when MSERs has more than one children. In short, given a MSERs tree, the procedure returns a set of disconnected nodes. The

```
1: procedure LINEAR-REDUCTION(T)
2:     if nchildren[T] = 0 then
3:         return T
4:     else if nchildren[T] = 1 then
5:         c ← LINEAR-REDUCTION(child[T])
6:         if var[T] ≤ var[c] then
7:             link-children(T, children[c])
8:             return T
9:         else
10:            return c
11:        end if
12:    else                                  ▷ nchildren[T] ≥ 2
13:        for each c ∈ children[T] do
14:            link-children(T, LINEAR-REDUCTION(c))
15:        end for
16:        return T
17:    end if
18: end procedure
```

Fig. 4. Linear reduction algorithm.

algorithm works as follows. For a given node t, tree accumulation checks the number of t's children; if t has no children, return t immediately; if t has more than two children, create an empty set C and append the result of applying tree accumulation to t's children to C; if one of the nodes in C has a lower variation than t's variation, return C, otherwise discard t's children and return t. Fig. 3(e) shows the result of applying tree accumulation to the tree shown in Fig. 3(d). Note that the final result is a set of disconnected nodes containing all the desired characters in the original MSERs tree.

3.1.5 Complexity Analysis

The linear reduction and tree accumulation algorithms effectively visit each node in the MSRE tree and do simple comparisons and pointer manipulations, thus the complexity is linear to the number of tree nodes. As MSERs' variations are already computed in the MSER extraction process, the computational complexity of variation regularization is mostly due to the calculations of MSERs'

```
1: procedure TREE-ACCUMULATION(T)
2:     if nchildren[T] ≥ 2 then
3:         C ← ∅
4:         for each c ∈ children[T] do
5:             C ← C ∪ TREE-ACCUMULATION(c)
6:         end for
7:         if var[T] ≤ min-var[C] then
8:             discard-children(T)
9:             return T
10:        else
11:            return C
12:        end if
13:    else                                  ▷ nchildren[T] = 0
14:        return T
15:    end if
16: end procedure
```

Fig. 5. Tree accumulation algorithm.

bounding rectangles, which are incrementally computable, as is discussed by Neumann and Matas [20].

3.2 Text Candidates Construction

3.2.1 Text Candidates Construction Algorithm Overview

Text candidates are constructed by clustering character candidates using single-link clustering. Intuitively, single-link clustering produces clusters that are elongated [29] and thus is particularly suitable for the text candidates construction task. Single-link clustering belongs to the family of hierarchical clustering; in hierarchical clustering, each data point is initially treated as a singleton cluster and clusters are successively merged until all points have been merged into a single remaining cluster. In the case of single-link clustering, the two clusters whose two closest members have the smallest distance are merged in each step. A distance threshold can be specified such that the clustering process is terminated when the distance between nearest clusters exceeds the threshold. The resulting clusters of single-link algorithm form a hierarchical cluster tree or cluster forest if termination threshold is specified. In our application of the single-link algorithm, each data point represents a character candidate and *top level* clusters in the final cluster tree (forest) correspond to text candidates.

The problem is of course how to determine the distance function and threshold for the single-link algorithm. We use the weighted sum of features as the distance function. Given two data points u, v, let $x_{u,v}$ be the feature vector characterizing the similarity between u and v, and the distance between u and v is defined as

$$d(u, v; w) = w^T x_{u,v}, \tag{3}$$

where w, the feature weight vector together with the threshold, can be learned using the proposed distance metric learning algorithm.

3.2.2 Feature Space

The feature vector $x_{u,v}$ is used to describe the similarities between data points u and v. Let x_u, y_u be the coordinates of the top left corner of u's bounding rectangle, h_u, w_u the height and width of the bounding rectangle of u, s_u the stroke width of u (stroke width computation is described in [28]), and $c1_u, c2_u, c3_u$ the average three channel color values of u, the feature vector $x_{u,v}$ includes the following features:

- Interval
$$\begin{cases} abs(x_v - x_u - w_u)/\max(w_u, w_v) & \text{if } x_u < x_v, \\ abs(x_u - x_v - w_v)/\max(w_u, w_v) & \text{else.} \end{cases}$$
- Width and height differences
$$abs(w_u - w_v)/\max(w_u, w_v), \;\; abs(h_u - h_v)/\max(h_u, h_v).$$
- Top and bottom alignments
$$\arctan\left(\frac{abs(y_u - y_v)}{abs(x_u + w_u/2 - x_v - w_u/2)}\right),$$
$$\arctan\left(\frac{abs(y_u + h_u - y_v - h_v)}{abs(x_u + w_u/2 - x_v - w_u/2)}\right).$$
- Color difference
$$\frac{\sqrt{(c1_u - c1_v)^2 + (c2_u - c2_v)^2 + (c3_u - c3_v)^2}}{255}.$$
- Stroke width difference
$$abs(s_u - s_v)/\max(s_u, s_v).$$

Note that these features are roughly in the same scale, and thus are not re-normalized in the learning process.

3.2.3 Distance Metric Learning

Many clustering algorithms rely on a good distance metric over the input space. One task of semi-supervised clustering is to learn a distance metric that satisfies the labels or constrains in the supervised data, given the clustering algorithm [30]–[32]. The strategy of metric learning is to learn the distance function by minimizing distance between point pairs in $\mathcal{C}$ while maximizing distance between point pairs in $\mathcal{M}$, where $\mathcal{C}$ specifies pairs of points in different clusters and $\mathcal{M}$ specifies pairs of points in the same cluster. In single-link clustering, clusters are formed by merging smaller clusters, the final resulting clusters will form a binary cluster tree, in which non-singleton clusters have exactly two direct subclusters. It is not hard to see that the following property holds for *top level* clusters: given the termination threshold ϵ, it follows that distances between subclusters of each top level cluster are less or equal to ϵ and distances between data pairs in different top level clusters are greater than ϵ, where the distance between clusters is defined as the distance between the closest members in each cluster. This property of single-link clustering enables us to design a learning algorithm that can learn the distance function and threshold simultaneously.

Given the top level cluster set $\{C_k\}_{k=1}^m$, we randomly initialize feature weights w and set $\mathcal{C}$ and $\mathcal{M}$ as

$$\mathcal{C} = \{(\hat{u}_k, \hat{v}_k) = \arg\min_{u \in C_k, v \in C_{-k}} d(u, v; w)\}_{k=1}^m, \tag{4}$$
$$\mathcal{M} = \{(u_k^*, v_k^*) = \arg\min_{u \in C_k^1, v \in C_k^2} d(u, v; w)\}_{k=1}^m, \tag{5}$$

where C_{-k} is the set of points excluding points in C_k; C_k^1 and C_k^2 are direct subclusters of C_k. Suppose ϵ is specified as the single-link clustering termination threshold. By the definition of single-link clustering, we must have

$$d(u, v; w) > \epsilon \text{ for all } (u, v) \in \mathcal{C}, \tag{6}$$
$$d(u, v; w) \leq \epsilon \text{ for all } (u, v) \in \mathcal{M}. \tag{7}$$

The above equations show that $\mathcal{C}$ and $\mathcal{M}$ can be corresponded as the positive and negative sample set of a classification problem, such that feature weights and threshold can be learned by minimizing the classification error. For this learning task, we use logistic regression as our model and by adopting the function of logistic regression, we define the following objective function,

$$J(\theta{:}\mathcal{C}, \mathcal{M}) = \frac{-1}{2m}\left(\sum_{(u,v)\in\mathcal{C}} \log\left(h_\theta(x'_{u,v})\right) + \sum_{(u,v)\in\mathcal{M}} \log\left(1 - h_\theta(x'_{u,v})\right)\right), \tag{8}$$

where

$$h_\theta(x'_{u,v}) = 1/(1 + \exp(-\theta^T x'_{u,v})), \tag{9}$$
$$\theta = \begin{pmatrix} -\epsilon \\ w \end{pmatrix}, \quad x'_{u,v} = \begin{pmatrix} 1 \\ x_{u,v} \end{pmatrix}.$$

Input: labeled clusters set $\{C_k\}_{k=1}^m$
Output: optimized θ such that J is minimized
Procedure:
randomly initialize θ
repeat
Stage 1: update $\mathcal{C}$ and $\mathcal{M}$ according to Equation (4) and (5), in which feature weights w are computed from current assignment of θ
Stage 2: given the positive sample set $\mathcal{C}$ and negative sample set $\mathcal{M}$, use logistic regression to learn the optimized parameter, update θ
until convergence or reach iteration limitation

Fig. 6. Self-training distance metric learning algorithm.

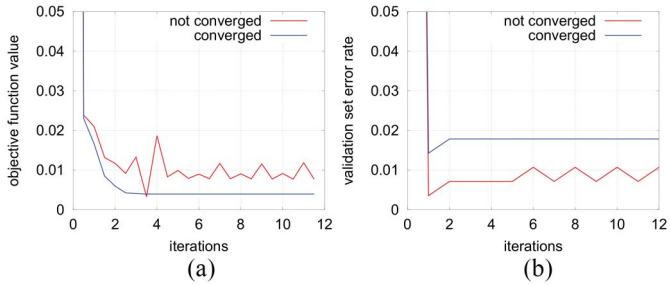


Fig. 7. Objective function value. (a) Validation set error rate of learned parameters. (b) After Stage 1 and Stage 2 in each iteration of two instances of the metric learning algorithm; the red line corresponds to the not converged instance and the blue line the converged one.

The weights w and threshold ϵ can be learned simultaneously by minimizing the objective function $J(\theta{:}\mathcal{M}, \mathcal{C})$ with respect to current assignments of $\mathcal{C}$ and $\mathcal{M}$

$$\theta^* = \arg\min_{\theta} J(\theta{:}\mathcal{C}, \mathcal{M}) \tag{10}$$

where the feature weights are required to be equal or more than zero. Minimization of this objective function is a typical nonlinear optimization problem and can be solved by classic gradient optimization methods [33].

Note that in the above scheme, an initial value of w has to be firstly specified in order to generate set $\mathcal{C}$ and $\mathcal{M}$ according to Equation (4) and (5). We design an iterative optimization algorithm where each iteration involves two successive steps for computation of $\mathcal{C}, \mathcal{M}$ and optimization with respect to $\mathcal{C}, \mathcal{M}$. We call this algorithm "*self-training distance metric learning*". Pseudocode for this learning algorithm is presented in Fig. 6. Given the top level cluster set $\{C_k\}_{k=1}^m$, the learning algorithm finds an optimized θ such that the objective function $J(\theta{:}\mathcal{C}, \mathcal{M})$ is minimized with respect to $\mathcal{C}, \mathcal{M}$. In this algorithm, one initial value for θ is set before the iteration begins; in the first stage of the iteration, $\mathcal{C}$ and $\mathcal{M}$ are updated according to Equation (4) and (5) with respect to current assignment of θ; in the second stage, θ is updated by minimizing the objective function with respect to the current assignment of $\mathcal{C}$ and $\mathcal{M}$. This two-stage optimization is then repeated until convergence or the maximum number of iterations is exceeded.

3.2.4 Empirical Analysis

Similar to most self-training algorithms, convergence of the proposed algorithm is not guaranteed because the objective function is not guaranteed to decrease in Stage 1. However, we find our algorithm can usually achieve good performance after a very small number of iterations. We label 466 texts in the ICDAR 2011 training database and use 70% of them as training data and 30% as validation data. Text in the same image corresponds to top level cluster set $\{C_k\}_{k=1}^m$ in Equation (4) and (5). In Stage 1, $\mathcal{C}$ and $\mathcal{M}$ of different images are computed separately but are merged to feed to the optimization precess in the next stage; in Stage 2, the objective function are optimized using the L-BFGS method [34] and the parameters are updated. Performance of the learned distance weights and threshold in Stage 2 is evaluated on the validation database in each iteration. Our experiments show that the learned parameters always have a very low error rate on the validation set after the first several iterations and no major improvement is observed in the continuing iterations. As a result, whether the algorithm converge or not has no great impact on the performance.

We plot the value of the objective function after Stage 1 and Stage 2 in each iteration of two instances (corresponding to a converged one and not converged one) of the algorithm in Fig. 7(a). The corresponding error rates of the learned parameters on the validation set in each iteration are plotted in Fig. 7(b). Notice that the value of the objective function and the validation set error rate drop immediately after the first several iterations. Fig. 7(b) shows that the learned parameters have different error rates due to different initial values, which suggests to run the algorithm several times to get the satisfactory parameters. The parameters for the single-link clustering algorithm in our scene text detection system are chosen based on the performance on the validation set.

3.3 Text Candidates Elimination

Using the text candidates construction algorithm proposed in Section 3.2, our experiment in ICDAR 2011 training database shows that only 9% of the text candidates correspond to true texts. As it is hard to train an effective text classifier using such an unbalanced database, most of the non-text candidates need to be removed before training the classifier. We propose to use a character classifier to estimate the posterior probabilities of text candidates corresponding to non-text and remove text candidates with high non-text probabilities.

The following features are used to train the character classifier: text region height, width and aspect ratio, smoothness (defined as the average difference of adjacent boundary pixels' gradient directions) and stroke width features (including mean and variance of character stroke widths). Characters with small aspect ratios such as "i", "j" and "l" are labeled as negative samples, as it is very uncommon that some words comprise many small aspect ratio characters.

Given a text candidate T, let $O(m, n; p)$ be the observation that there are m ($m \in \mathbb{N}, m \geq 2$) character candidates in T, where n ($n \in \mathbb{N}, n \leq m$) candidates are classified as non-characters by a character classifier of accuracy (on the validation set) p ($0 < p < 1$). The probabilities of the observation conditioning on T corresponding

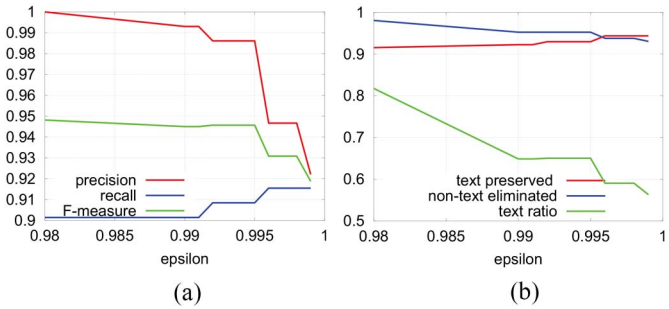


Fig. 8. Performance of different ε on the validation set. (a) Precision, recall and f-measure of text classification task. (b) Ratio of preserved text samples, ratio of eliminated non-text samples and ratio of text samples.

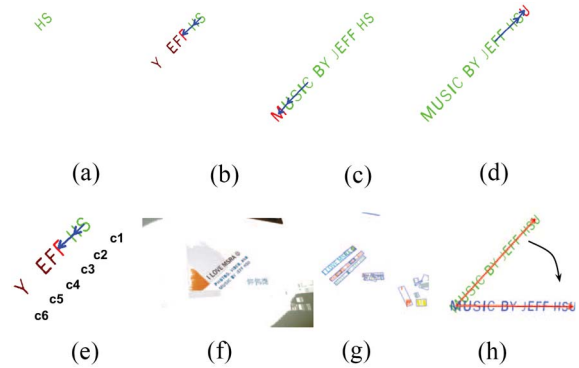


Fig. 9. Forward-backward algorithm demonstration.

to text and non-text are $P(O(m,n;p)|\text{text}) = p^{m-n}(1-p)^n$ and $P(O(m,n;p)|\text{non-text}) = (1-p)^{m-n}p^n$ respectively. Let $P(\text{text})$ and $P(\text{non-text})$ be the prior probability of T corresponding to text and non-text. By applying Bayes' rule, the posterior probability of T corresponding to non-text given the observation is

$$P(\text{non-text}|O(m,n;p)) = \frac{P(O(m,n;p)|\text{non-text})P(\text{non-text})}{P(O(m,n;p))}, \quad (11)$$

where $P(O(m,n;p))$ is the probability of the observation

$$P(O(m,n;p)) = P(O(m,n;p)|\text{text})P(\text{text}) + P(O(m,n;p)|\text{non-text})P(\text{non-text}), \quad (12)$$

The candidate region is rejected if

$$P(\text{non-text}|O(m,n;p)) \geq \varepsilon, \quad (13)$$

where ε is the threshold.

Our experiment shows that text candidates of different lengths (the number of characters) tend to have different probabilities of being text. For example, on the ICDAR 2011 training set, 1.25% of text candidates of length two correspond to text, while 30.67% of text candidates of length seven correspond to text, which suggests to set different priors for text candidates of different length. Given a text candidate T of length s, let N_s be the total number of text candidates of length s, N_s^* be the number of text candidates of length s that correspond to text, we estimate the prior of T being text as $P_s(\text{text}) = N_s^*/N_s$, and the prior of T being non-text as $P_s(\text{non-text}) = 1 - P_s(\text{text})$. If no text candidates of length s are found in the training database, $P_s(\text{text})$ is set to a default value of 0.5. These priors are computed based on statistics on the ICDAR training database.

To find the appropriate ε in Equation (13), we use 70% of ICDAR training database to train the character classifier and the text classifier, the remaining 30% as the validation set to test the performance of different ε. Fig. 8(a) shows the precision, recall and f-measure of text candidates classification task on the validation set. As ε increases, text candidates are more unlikely to be eliminated, results in the increase of recall value. For the scene text detection task, recall is preferred over precision, until $\varepsilon = 0.995$ is reached, where a major decrease of precision and f-measure occurred, which can be explained by the sudden decrease of ratio of text samples (see Fig. 8(b)). Fig. 8(b) shows that at $\varepsilon = 0.995$, 92.95% of text are preserved, while 95.25% of non-text are eliminated.

3.4 Extension to Multi-Orientation Text Detection

Arbitrary orientation scene text detection is much more complicated than horizontal text detection. The fundamental difficulty is that the text line alignment feature can no longer be used to regularize the text construction process. However, the text line alignment feature is always there – text are still aligned, but in different orientations. Arbitrary orientation alignment feature cannot be used in a clustering-based text construction algorithm, as the orientation is not fixed and the clustering algorithm is not able to see the whole picture until the clustering process terminates. Consequently, we design an arbitrary orientation text detection algorithm using a heuristic strategy – the *forward-backward* algorithm. After we have detected all text lines using the forward-backward algorithm, we compute the orientations of text lines and convert arbitrary orientation text lines into horizontal text lines (Fig. 9(h)). Converted text lines are again fed into our horizontal text line detection pipeline (text candidates construction, elimination and classification) to be detected, which constructs our whole multi-orientation text detection system.

We give a simple description of our algorithm here. After character candidates extraction (in Section 3.1), we first sort the (unordered) pairs of components according to their priorities. (char, char) pair is at the highest priority rank, (non-char, char) pair is at the medium priority rank and (non-char, non-char) pair is at the lowest priority rank. Here the character identity is estimated by a character classifier (in Section 3.3). Since two component pairs are in the same priority rank, the component pair with a smaller inter-component distance has a higher priority. After we have sorted the sequence of component pairs, the forward-backward algorithm begins with the component pair (c1, c2) enjoying the highest priority (Fig. 9(a)) and tries to expand to the forward direction (c1 $\rightarrow$ c2) (Fig. 9(b)). If the forward direction is no longer expandable (Fig. 9(c)), we turn to the opposite direction and try to expand backward (Fig. 9(d)). After the backward direction is no longer expandable (Fig. 9(d)), we have finished the detection of one text line (Fig. 9(d)). We erase component pairs containing any components appearing in the detected text line from the component pair sequence. We then begin detecting

TABLE 1
Performance (%) Comparison of Text Detection Algorithms on ICDAR 2011 Testing Database

Method	Recall	Precision	f
Our Method	**68.26**	**86.29**	**76.22**
Shi et al.'s method [23]	63.1	83.3	71.8
Kim's Method [41]	62.47	82.98	71.28
Neumann and Matas [20]	64.7	73.1	68.7
Yi's Method	58.09	67.22	62.32
TH-TextLoc System	57.68	66.97	61.98
Neumann's Method	52.54	68.93	59.63

of next text line by starting off with the next highest priority component pair. Component expansion involves two steps: component candidates searching and optimal component candidate selecting. To be qualified for a candidate, a component should meet the angle requirement and the distance requirement. Consider the situation depicted in Fig. 9(e). We are trying to expand from component c2 and the component under consideration is c3. The angle requirement could be roughly stated as: the angle between baseline Euclidean vector (represented by the blue arrow line from c1 to c2) and the Euclidean vector from c2 to c3 (represented by the blue arrow line from c2 to c3) should not exceed a threshold; the distance requirement refers to that the distance between c2 and c3 should not exceed a threshold. After we have found all the candidates (c3-c6 in this case), an optimal component (c3 in this case) is selected from candidates using the following two criterion: (1) the optimal candidate should be as close to the current component as possible, (2) the line between centroids of the current component and the optimal candidate should not intersect with other candidates. On the other hand, if we have not found any component candidates fulfilling the angle and distance requirements, the current component is not expandable and expanding on this direction is finished.

In the forward-backward algorithm, we are still using the text line alignment feature. The direction of alignment is calculated from the first pair of components (Fig. 9(a)), and is continually recalculated and used to regularize the expanding operation. The priority scheme ensures that the first pair of components can be trusted (to be characters) to calculate the initial alignment direction. The threshold for the angle requirement is specified empirically. Component distance and threshold use our learned distance metrics in Section 4.5.2, with the exception that top and bottom alignment feature values are set to 0, since we are not considering them anymore.

4 Experimental Results

In this section, first we compare our technology[1] with several state-of-the-art methods on a variety of public databases. Then, we present experiments on components (MSERs pruning, distance metric learning and text candidates elimination) of the proposed method. Please note that without specification, we use the definitions in ICDAR 2011 competition [17] for text detection precision, recall and f-measure calculation; and the speed (s) for each scheme of our technology is profiled on a Linux laptop with a 2.00GHZ processor.

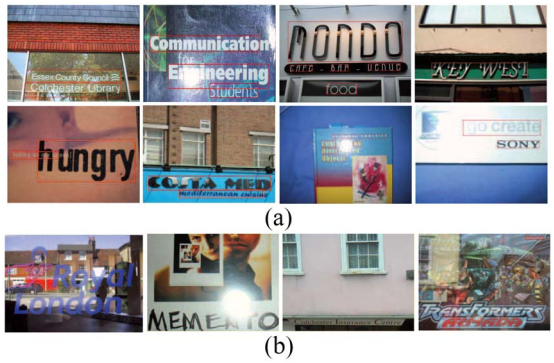


Fig. 10. Text detection samples of our system on ICDAR 2011 database. (a) Successful. (b) Failed samples.

4.1 Experiments on ICDAR 2011 Databases

The ICDAR 2011 Robust Reading Competition (Challenge 2: Reading Text in Scene Images) database [17] is a widely used database[2] for benchmarking scene text detection algorithms. The database contains 229 training images and 255 testing images. The proposed system is trained on the training set of this database. In order to measure the performance of the proposed system on this database, text candidates identified as text by our system are further partitioned into words by classifying inner character distances into character spacings and word spacings using an AdaBoost classifier [28].

Table 1 shows the performance of our system, Neumann and Matas' method [20], a very recent MSER-based method by Shi *et al.* [23] and some of the top scoring methods (Kim's method, Yi's method, TH-TextLoc system and Neumann's method) from ICDAR 2011 Competition. As can be seen from Table 1, our method produces much better recall, precision and f-measure over other methods on this database. Note that the first four methods in Table 1 are all MSER-based methods. Apart from the detection quality, the proposed system offers speed advantage over some of the listed methods. The average processing speed of the proposed system is 0.43s per image. The speed of Shi *et al.*'s method [23] on a PC with a 2.33GHZ processor is 1.5s per image. The speed of Neumann and Matas' method [20] (including text localization and recognition) is 1.8s per image on a "standard PC". Fig. 10(a) shows some texts successfully detection by our system on ICDAR 2011 database. Fig. 10(b) shows some scene texts that our system fails to detect due to very complex background, seriously nonuniform illumination (with reflective surfaces), highly blurred text and unusual fonts.

The ICDAR 2011 Robust Reading Competition consists of two tracks. Apart from the natural scene images track (Challenge 2), we also profile our system on the born-digital images track (Challenge 1) [35]. Though born-digital images are not scene images, they have similar challenging issues for text detection, and detecting texts in such images is also an important real application. Compared to scene images,

1. An online demo of our scene text detection system is available at *http://prir.ustb.edu.cn/TexStar/scene-text-detection/*.

2. The ICDAR 2011 Robust Reading Competition dataset (Challenge 2) is available at *http://robustreading.opendfki.de/wiki/SceneText*.

TABLE 2
Performance (%) of Our Text Detection System on ICDAR 2011 Competition Born-Digital Image Database (Challenge 1), Where "Textorter" is the Winning Method in the 2011 Competition

Method	Recall	Precision	f
Our Method	84.21	93.49	88.61
TCC_textDetector	88.64	81.46	84.90
Anthimopoulos	81.88	87.35	84.53
Textorter	69.08	85.54	76.43

Fig. 11. Text detection examples by our system on the multilingual database (failed samples in the last row).

born-digital images are of low resolution and texts are digitally created on the images; the backgrounds are usually clean and there are no serious illumination problems. In order to deal with the low-resolution problem, images are amplified seven times over original images before processing, which is empirically estimated on the training set. The system used is completely the same as the one used in the scene image database. Table 2 presents the performances of our system and some top-scoring methods on this database[3]. Experimental results definitely show that our proposed method has a very surprised performance.

4.2 Experiments on Multilingual Database

The multilingual database (including Chinese and English texts, see Fig. 11) is initially published by Pan *et al*. [14] to evaluate the performance of their scene text detection system[4]. The training set contains 248 images and the testing set contains 239 images. As there are no apparent spacing between Chinese words, this multilingual database only provides ground truths for text lines. We hence evaluate the text line detection performance of our system without further partitioning text into words. Fig. 11 shows some detection examples by our system.

The performances of our systems (with the character classifier trained on the multilingual training database, see first row of Table 4) and Pan *et al*.'s method [14] are presented in Table 3. The ICDAR 2003 competition evaluation scheme is used for fair comparison. As can be seen from Table 3, our method performs much better than Pan *et al*.'s method on this database both in recall and precision. In addition, our system runs much faster than Pan *et al*.'s method, which is profiled on a PC with a 3.4GHz processor.

3. These results are publicly available at *http://www.cvc.uab.es/icdar2011competition/?com=results*, where "TexStar" is our method.

4. *http://onedrive.live.com/?cid=DDDA0698&id=DDDA0698413AC264!283*.

TABLE 3
Performance (%) of Different Text Detection Algorithms on the Multilingual Database

Method	Recall	Precision	f	Speed (s)
Our method	68.5	82.6	74.6	0.22
Pan et al.'s method [14]	65.9	64.5	65.2	3.11

TABLE 4
Performance of Different Experimental Configurations on ICDAR and Multilingual Databases

Database	Character classifier trained on	recall, precision, f
Multilingual test set	**Multilingual train set**	**68.5, 82.6, 74.6**
	ICDAR 2011 train set	63.2, 79.4, 70.4
	ICDAR 2011 train set + Multilingual train set	67.5, 78.3, 72.5
ICDAR 2011 test set	Multilingual train set	66.02, 85.14, 74.37
	ICDAR 2011 train set	**68.26, 86.29, 76.22**
	ICDAR 2011 train set + Multilingual train set	67.30, 85.44, 75.29

In order to examine the impaction of the character classifier to the overall system performance, we design several experimental configurations (shown in Table 4). ICDAR 2011 database contains only English characters, while the multilingual database contains much Chinese characters and some English characters. It can be seen from Table 4 that the character classifier trained on most relevant, most irrelevant and mixed databases makes best, worst and moderate performance respectively. A highly reliable character classifier is very important for the overall system performance.

The samples in the last row in Fig. 11 show some texts our system fails to detect. Texts can be missed in the text candidates elimination stage and text classification stage. Unlike English characters, the number of Chinese characters is huge and they can be very different from each other in appearance. This brings up a great challenge for training of the effective character and text classifiers. In addition, the strokes of Chinese characters can be disconnected, which could also lead to troubles in character candidates grouping.

4.3 Experiments on Street View Database

We also profile our system on the database with 307 natural images[5] proposed by Epshtein *et al*. [9], which is even harder than the ICDAR database. Most of the images in this database are street view images and they contain many repeating structures such as windows and bricks. Some images are captured in a low light environment and texts tend to be small, blurred and of low contrast (see Fig. 13). We find that our character classifier trained on ICDAR database is struggling to identify many low quality characters, which could lead to misjudgments in the text candidates elimination stage. As we can no longer safely assume that our classifier can achieve a high accuracy ($\geq$ 0.9) in Equation (13) on this database, we profilethe system

5. *http://research.microsoft.com/en-us/um/people/eyalofek/text_detection_database.zip*.

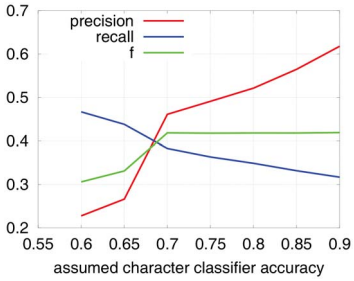


Fig. 12. System performance under different character classifier accuracy assumption in Equation (13).

TABLE 5
Performance on Epshtein *et al.*'s Database

Method	Recall	Precision	f
Our method, with character classifier trained on ICDAR 2011 train set	0.32	0.62	0.42
Our method, with character classifier trained on street view database	0.41	0.66	0.51
Epshtein et al.'s method [9]	0.42	0.54	0.47
Phan et al.'s method [36]	0.51	0.50	0.51

performance under different character classifier accuracy assumption. It can be seen from Fig. 12 that as we increase our confidence on the classifier, recall decreases and precision increases, but the f-measure tends to have fixed values.

We set $p = 0.9$ for our system. It's a limitation of our system relying on trained character and text classifiers on one database, which could lead to troubles when we face a quite different database. Consequently, we use the street view text dataset[6] proposed by Wang *et al*. [37] (the whole database) to train our character classifier and profile the system on Epshtein *et al*.'s database. The method proposed by Epshtein *et al*. [9] with Stroke Width Transform (SWT) is language neutral and does not involve a classifier for character and text identification. This SWT algorithm has been used in several recent research works [9], [38] and implemented in a popular computer vision library the libccv[7]. However, their method involves selecting about 20 parameters, which is a challenging, time-consuming, and error-prone task.

In Table 5, we present the performance of three methods, our method, Epshtein *et al*.'s method [9], and Phan *et al*.'s method [36] which also involves empirically selecting many parameters. Experimental results show our technology has a very encouraged performance if just a new character classifier is trained on the new database. Fig. 13 shows some detection samples on this challenging database. As observed, it is a challenge for our method to detect highly blurred low-resolution texts in natural scenes. This is however also one open problem for almost all the approaches in the field and hence deserves our attention in the future.

6. *http://vision.ucsd.edu/~kai/svt/*
7. The libccv is available at *http://libccv.org*.

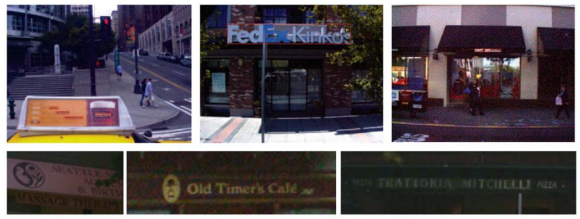


Fig. 13. Text detection samples of our system on the database proposed by Epshtein *et al.* [9]. (1) Successful (1^{st} row). (2) Unsuccessful samples (2^{nd} row).

TABLE 6
Performance on the Multi-Orientation Database

Method	Recall	Precision	f	Speed (s)
Our method	0.61	0.71	0.66	0.8
Yao et al. [38]	0.63	0.63	0.60	7.2

4.4 Experiments on Multi-Orientation Database

We evaluate our multi-orientation text detection system on the MSRA-TD500 database[8] using the evaluation protocol proposed by Yao *et al*. [38]. As can be seen from Table 6, the overall performance of our method is better than Yao *et al*.'s method though our recall is slightly lower than their method. Unsuccessful detection (see Fig. 14) may be due to the following reasons: (1) The character classifier is not able to identify some challenging characters; and (2) the forward-backward algorithm is unable to handle some multi-orientation text scenarios where characters in multiple text lines have a very similar structure and can be easily grouped as a vertical line. In addition, our system runs almost an order of magnitude faster than Yao *et al*.'s method, which is profiled on a machine with a 2.53GHz processor.

Considering that multi-orientation detection on horizontal texts may produce more false positives (by linking character candidates in all directions), we also apply our multi-orientation text detection system on several horizontal and near horizontal text databases. As can be seen from Table 7, all experimental results show that our multi-orientation detection system has a very competitive performance compared to other state-of-the-art methods. By the way, compared with horizontal-only detection, though the recall performance is similar, the precision degradation of the multi-orientation system is obvious. It would be more robust by combining direction voting and selection in line linking for multi-oriented text lines, which is a near future topic of our research.

4.5 System Components Experiments

4.5.1 Experiments on Character Candidates Extraction

In this section we present the performance comparison of three character extraction algorithms: our MSER-based pruning algorithm, the latest pruning method proposed by Neumann and Matas (the ER-based algorithm in [20]) and the SWT algorithm [9]. As our MSER-pruning algorithm is built on the MSERs extraction algorithm, we also examine the impaction of MSERs parameters.

8. This dataset is available at *http://www.iapr-tc11.org/mediawiki/index.php/MSRA_Text_Detection_500_Database_(MSRA-TD500)*.

Fig. 14. Text detection samples on multi-orientation database. The last two images are failed samples.

TABLE 7
Performance of Our Horizontal and Multi-Orientation Text Detection Systems on Horizontal or Near Horizontal Text Databases

Database	System	Recall, Precision, f	Speed
ICDAR 2011 test database	horizontal	0.6826, 0.8629, 0.7622	0.43
	multi-oriented	0.6663, 0.8200, 0.7352	0.95
Multilingual database [14]	horizontal	0.685, 0.826, 0.746	0.22
	multi-oriented	0.678, 0.687, 0.683	0.47
Epshtein's database [9]	horizontal	0.41, 0.66, 0.51	0.42
	multi-oriented	0.42, 0.54, 0.47	0.60

We use two metrics, character-level recall and speed, to compare the performance of different character extraction algorithms. The definition of character-level recall is similar to the recall definition in ICDAR 2003 competition [39], except that the bounding rectangle is character level rather than text level. We first compare the performance of our proposed algorithm with the SWT algorithm [9][9] on the ICDAR 2011 training set. The character-level recalls of our MSER-base algorithm and the SWT algorithm are **88.52**% and **69.54**% respectively. The average speeds of our text detection system and the SWT system are 444.36 and 694.88 ms/image respectively. These results show that our MSER-pruning algorithm performs much better than SWT both in extraction quality and speed.

In Neumann and Matas's work [20], character-level recall is defined as the ratio of detected character among ground-truth characters, where a character is considered as detected if over 90% of the area of its bounding rectangle is matched by the bounding rectangle of some detected character. The recall value of our algorithm under this new definition, along with recall of the ER-based method (speed is not available from their work) are presented in Table 8. It can be seen that the recall rate of our method is better than the ER-based method both in the single-channel scheme and the multi-channels scheme. The multi-channels recall rate of both methods is significantly better than the single-channel recall rate, but the computational cost of multi-channels can be several times bigger than single-channel's. Our method and the ER-based method both prune MSERs by maximizing local character probabilities. However, our algorithm uses simple regularized variations while the ER-based method uses a complex classifier, where features like Euler number and horizontal crossing are language-dependent.

The proposed character extraction algorithm is built on extracted MSERs and the MSERs extraction algorithm is controlled by several parameters: Δ controls how the variation is calculated; maximal variation v_+ excludes too unstable MSERs; minimal diversity d_+ removes duplicate MSERs by measuring the size difference between a MSER and its parent. A conservative parameter setting (lower Δ, higher v_+ and lower d_+) enables the MSERs algorithm to detect low contrast characters (due to v_+) without missing regions that are more likely corresponding to characters (due to Δ and d_+). In our system we use a more conservative parameter setting ($\Delta = 1, v_+ = 0.5, d_+ = 0.1$) than the default one ($\Delta = 5, v_+ = 0.25, d_+ = 0.2$) in [40]. $\Delta = 1$ is basically the smallest value one can set for Δ. The other two parameters are empirically estimated with several low quality images. Our experiments on ICDAR 2011 training database show that our parameter setting detects much more MSERs (from 460 to 2017) and achieves a considerable increase in the recall value (89.68% to 93.00%), while the speed loss is not significant (from 352.6 ms/image to 381.5 ms/image). However, directly extracted MSERs are repeated and cannot be used directly. In the case of our parameter setting, the MSERs pruning algorithm could remove 65.8% MSERs, incurring a decrease of recall from 93.00% to 88.52%. The pruning time is not significant compared with the MSERs extraction time (32.4 ms/image vs. 381.5 ms/image).

TABLE 8
Character-Level Recall of Our MSER-Based Algorithm and ER-Based Algorithm

Algorithm	Character-level Recall (%)
Our algorithm	90.2 (grayscale) 95.2 (B, G, R, three channels)
ER-based algorithm	85.6 (single channel) 93.7 (I, H, S, I Gradient, four channels)

To examine closely the MSERs prunning algorithm, we show some failed samples in Fig. 15. There are two cases of bad segmentation: multiple characters being segmented as one character (the first three samples in Fig. 15) and one character being segmented as multiple characters (the last sample in Fig. 15). The first case could be due to indistinct character boundaries and inherently inseparable characters. The second case could be caused by seriously nonuniform color distribution in one character. How to further refine the performance of the engaged MSERs pruning algorithm is an interesting topic. We will leave this as future work.

4.5.2 Experiments on Learned Distance Function

Table 9 shows the system performance using learned and hand-tuned distance parameters. We spend quite a long time for empirically and manually tuning distance weights

Fig. 15. Failed segmentation samples. Characters' boundaries are labeled red. Bad segmented characters are labeled with blue bounding rectangles.

9. The SWT executable is adopted from libccv.

TABLE 9
Overall System Performance on ICDAR 2011 Testing Set with Learned and Hand-Tuned Distances

Scheme	Recall	Precision	f
Learned weights	68.26	86.29	76.22
Hand-tuned weights	68.02	84.82	75.50

in text candidates construction based on ICDAR 2011 training set before we propose the distance metric learning algorithm. Even though, learned parameters perform better than carefully hand-tuned ones. Moreover, preparing training data for metric learning by specifying must-link and cannot-link character candidates is much easier than manually tuning the distance function.

The learned and hand-tuned distance function parameters are presented in Table 10. Hand-tuned parameters are renormalized with respect to the learned threshold value for the convenience of comparison. The learned parameters are rather different from hand-tuned parameters. The bottom alignment and stroke width feature are the most useful features. Compared to the bottom alignment feature, the top alignment feature has not been playing an important role as we have expected, but this difference conforms to the fact that text is usually well bottom aligned but not well top aligned. The effect of the interval feature is underestimated and the effect of the color difference feature are overestimated, but they both don't play important roles among the learned weights. Please note that learned parameters have small changes from time to time when we run our metric learning algorithm, but no significant variation of the overall system performance is observed.

4.5.3 Experiments on Text Candidates Elimination

The text candidates elimination algorithm could affect the system performance in two ways by: (1) eliminating non-text candidates, (2) creating a balanced database (by eliminating up to 95% non-text, see Section 3.3) so that we can train a more powerful text classifier. As the text classifier trained in the last stage is closely associated with the elimination operation, we design two schemes to examine the effects of text candidates elimination: (1) *Scheme-I*, where text candidates elimination is performed, and the text classifier is trained on the database after candidates elimination; and (2) *Scheme-II*, where the text classifier is directly trained on the database without candidates elimination.

As can be seen from Table 11, compared with Scheme-I, Scheme-II's recall and precision both decrease, and the precision decrease is much serve than the recall decrease. This could be explained that there is not a text candidates elimination step in Scheme-II, and without this step, too many non-text candidates are left to the classifier, which could lead to not only a large decrease in precision but also a little decrease in recall. Actually, in Scheme-II, the text classifier is trained on a seriously unbalanced database where negative samples (non-text) are majority.

TABLE 10
Hand-Tuned and Learned Distance Function Parameters, Where "Diff" and "Bott" Represent "Difference" and "Bottom"

Params	threshold	interval	width diff	height diff
Hand-tuned	21.52	0.85	12.82	7.68
Learned	21.52	3.78	9.84	2.12
Params	top align	bott align	color diff	stroke diff
Hand-tuned	6.40	38.40	40.82	32.02
Learned	1.25	21.71	1.22	21.57

TABLE 11
System Performance on ICDAR 2011 Testing Set with and Without Text Candidates Elimination

Scheme	Recall	Precision	f
Scheme-I	68.26	86.29	76.22
Scheme-II	65.57	77.49	71.03

5 Conclusion

This paper presents a new MSER-based scene text detection method with several novel techniques. First, we propose a fast and accurate MSERs pruning algorithm that enables us to detect most characters even when the image is in low quality. Second, we propose a novel self-training distance metric learning algorithm that can learn distance weights and clustering threshold simultaneously; text candidates are constructed by clustering character candidates by the single-link algorithm using the learned parameters. Third, we put forward to use a character classifier to estimate the posterior probability of text candidate corresponding to non-text and eliminate text candidates with high non-text probability, which helps to build a more powerful text classifier. Finally, by integrating the above new techniques, we build a robust scene text detection system that exhibits superior performance over state-of-the-art methods on a variety of public databases. Our text detection system ("USTB_TexStar") also won the first place of both "Text Localization in Real Scenes" and "Text Localization in Born-Digital Images" in the ICDAR 2013 Robust Reading Competition [42].

We empirically analyze several main limitations of our technology for further research, which are also open issues in the literature. First, how to detect highly blurred texts in low-resolution natural scene images (as shown in Fig. 10 and 13) is a near future issue. Second, some multilingual texts (e.g., Chinese, as shown in Fig. 11) have quite different characteristics from English texts. Consequently, adaptively detecting a variety of multilingual texts simultaneously is also a challenge. Third, how to detect multi-orientation texts needs to be further investigated, especially for similar multiple text lines with a seriously skewed distortion.

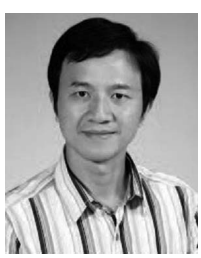
Acknowledgments

We are grateful to Prof. B. Xiao and Dr. H. Man for helpful discussions, and to the anonymous reviewers for their constructive comments. The research is partly supported by National Basic Research Program of China (2012CB316301) and National Natural Science Foundation of China (61105018, 61175020).

References

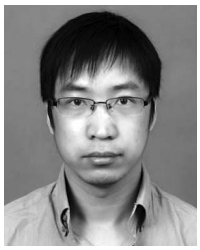
[1] Y. Zhong, H. Zhang, and A. Jain, "Automatic caption localization in compressed video," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 4, pp. 385–392, Apr. 2000.

[2] H. Li, D. Doermann, and O. Kia, "Automatic text detection and tracking in digital video," *IEEE Trans. Image Process.*, vol. 9, no. 1, pp. 147–156, Jan. 2000.
[3] J. Weinman, E. Learned-Miller, and A. Hanson, "Scene text recognition using similarity and a lexicon with sparse belief propagation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 10, pp. 1733–1746, Oct. 2009.
[4] X.-C. Yin, H.-W. Hao, J. Sun, and S. Naoi, "Robust vanishing point detection for mobile cam-based documents," in *Proc. ICDAR*, Beijing, China, 2011, pp. 136–140.
[5] P. Shivakumara, Q. P. Trung, and C. L. Tan, "A laplacian approach to multi-oriented text detection in video," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 2, pp. 412–419, Feb. 2011.
[6] X. Chen and A. Yuille, "Detecting and reading text in natural scenes," in *Proc. IEEE Conf. CVPR*, vol. 2. Washington, DC, USA, 2004, pp. 366–373.
[7] J.-J. Lee, P.-H. Lee, S.-W. Lee, A. Yuille, and C. Koch, "Adaboost for text detection in natural scene," in *Proc. ICDAR*, Beijing, China, 2011, pp. 429–434.
[8] K. Kim, K. Jung, and J. Kim, "Texture-base approach for text detection in images using support vector machines and continuously adaptive mean shift algorithm," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 12, pp. 1631–1639, Dec. 2003.
[9] B. Epshtein, E. Ofek, and Y. Wexler, "Detecting text in natural scenes with stroke width transform," in *Proc. IEEE Conf. CVPR*, San Francisco, CA, USA, 2010, pp. 2963–2970.
[10] C. Yi and Y. Tian, "Text string detection from natural scenes by structure-based partition and grouping," *IEEE Trans. Image Process.*, vol. 20, no. 9, pp. 2594–2605, Sept. 2011.
[11] C. Mancas-Thillou and B. Gosselin, "Color text extraction with selective metric-based clustering," *Comput. Vis. Image Und.*, vol. 107, no. 1–2, pp. 97–107, 2007.
[12] C. Yi and Y. Tian, "Localizing text in scene images by boundary clustering, stroke segmentation, and string fragment classification," *IEEE Trans. Image Process.*, vol. 21, no. 9, pp. 4256–4268, Sept. 2012.
[13] C. Yi and Y. Tian, "Text extraction from scene images by character appearance and structure modeling," *Comput. Vis. Image Und.*, vol. 117, no. 2, pp. 182–194, 2013.
[14] Y.-F. Pan, X. Hou, and C.-L. Liu, "A hybrid approach to detect and localize texts in natural scene images," *IEEE Trans. Image Process.*, vol. 20, no. 3, pp. 800–813, Mar. 2011.
[15] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proc. Int. Conf. Mach. Learn.*, San Francisco, CA, USA, 2001, pp. 282–289.
[16] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide baseline stereo from maximally stable extremal regions," in *Proc. Brit. Mach. Vis. Conf.*, vol. 1. 2002, pp. 384–393.
[17] A. Shahab, F. Shafait, and A. Dengel, "ICDAR robust reading competition challenge 2: Reading text in scene images," in *Proc. ICDAR*, 2011, pp. 1491–1496.
[18] H. Chen *et al.*, "Robust text detection in natural images with edge-enhanced maximally stable extremal regions," in *Proc. IEEE Int. Conf. Image Process.*, 2011, pp. 2609–2612.
[19] C. Merino-Gracia, K. Lenc, and M. Mirmehdi, "A head-mounted device for recognizing text in natural scenes," in *Proc. Int. Workshop CBDAR*, Beijing, China, 2011, pp. 29–41.
[20] L. Neumann and J. Matas, "Real-time scene text localization and recognition," in *Proc. IEEE Conf. CVPR*, Providence, RI, USA, 2012, pp. 3538–3545.
[21] L. Neumann and J. Matas, "Text localization in real-world images using efficiently pruned exhaustive search," in *Proc. ICDAR*, Beijing, China, 2011, pp. 687–691.
[22] L. Neumann and J. Matas, "A method for text localization and recognition in real-world images," in *Proc. ACCV*, vol. 3. Queenstown, New Zealand, 2011, pp. 770–783.
[23] C. Shi, C. Wang, B. Xiao, Y. Zhang, and S. Gao, "Scene text detection using graph model built upon maximally stable extremal regions," *Pattern Recognit. Lett.*, vol. 34, no. 2, pp. 107–116, 2013.
[24] K. Jung, K. Kim, and A. Jain, "Text information extraction in images and video: A survey," *Pattern Recognit.*, vol. 37, no. 5, pp. 977–997, 2004.
[25] J. Liang, D. Doermann, and H. Li, "Camera-based analysis of text and documents: A survey," *IJDAR*, vol. 7, no. 2–3, pp. 84–104, 2005.
[26] J. Zhang and R. Kasturi, "Extraction of text objects in video documents: Recent progress," in *Proc. IAPR Workshop DAS*, Nara, Japan, 2008, pp. 1–13.
[27] F. Yin and C.-L. Liu, "Handwritten Chinese text line segmentation by clustering with distance metric learning," *Pattern Recognit.*, vol. 42, no. 12, pp. 3146–3157, 2009.
[28] X. Yin, X.-C. Yin, H.-W. Hao, and K. Iqbal, "Effective text localization in natural scene images with MSER, geometry-based grouping and ada boost," in *Proc. Int. Conf. Pattern Recognit.*, Tsukuba, Japan, 2012, pp. 725–728.
[29] A. Jain, M. Murty, and P. Flynn, "Data clustering: A review," *ACM Comput. Surv.*, vol. 31, no. 3, pp. 264–323, 1999.
[30] M. Bilenko, S. Basu, and R. J. Mooney, "Integrating constraints and metric learning in semi-supervised clustering," in *Proc. Int. Conf. Mach. Learn.*, Banff, AB, Canada, 2004, pp. 81–88.
[31] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell, "Distance metric learning, with application to clustering with side-information," in *Advances in Neural Information Processing Systems 15*. MIT Press, 2002, pp. 505–512.
[32] D. Klein, S. D. Kamvar, and C. D. Manning, "From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering," in *Proc. Int. Conf. Mach. Learn.*, San Francisco, CA, USA ,2002, pp. 307–314.
[33] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. Springer-Verlag, 2009.
[34] J. Nocedal, "Updating Quasi-Newton matrices with limited storage," *Math. Comput.*, vol. 35, no. 151, pp. 773–782, 1980.
[35] D. Karatzas, S. R. Mestre, J. Mas, F. Nourbakhsh, and P. P. Roy, "ICDAR 2011 robust reading competition challenge 1: Reading text in born-digital images (web and email)," in *Proc. ICDAR*, 2011, pp. 1485–1490.
[36] T. Q. Phan, P. Shivakumara, and C. L. Tan, "Detecting text in the real world," in *Proc. ACM Int. Conf. MM*, New York, NY, USA, 2012, pp. 765–768.
[37] K. Wang, B. Babenko, and S. Belongie, "End-to-end scene text recognition," in *Proc. IEEE Int. Conf. Comput. Vis.*, Barcelona, Spain, 2011, pp. 1457–1464.
[38] C. Yao, X. Bai, W. Liu, Y. Ma, and Z. Tu, "Detecting texts of arbitrary orientations in natural images," in *Proc. IEEE Conf. CVPR*, Providence, RI, USA, 2012, pp. 1083–1090.
[39] S. Lucas *et al.*, "ICDAR 2003 robust reading competitions: Entries, results and future directions," *IJDAR*, vol. 7, no. 2–3, pp. 105–122, 2005.
[40] A. Vedaldi and B. Fulkerson, *VLFeat: An Open and Portable Library of Computer Vision Algorithms* [Online]. Available: http://www.vlfeat.org/, 2008.
[41] H. I. Koo and D. H. Kim, "Scene text detection via connected component clustering and nontext filtering," *IEEE Trans. Image Process.*, vol. 22, no. 6, pp. 2296–2305, June 2013.
[42] D. Karatzas *et al.*, "ICDAR 2013 robust reading competition," in *Proc. ICDAR*, Washington, DC, USA, 2013, pp. 1115–1124.



Xu-Cheng Yin (M'10) received the B.Sc and M.Sc. degrees both in computer science from the University of Science and Technology Beijing, Beijing, China, in 1999 and 2002, respectively, and the Ph.D. degree from the Institute of Automation, Chinese Academy of Sciences, China, in 2006. He is an Associate Professor with the Department of Computer Science and Technology, University of Science and Technology Beijing. Currently, from January 2013 to January 2014, he was a Visiting Researcher with the Center for Intelligent Information Retrieval, University of Massachusetts Amherst, Amherst, MA, USA. From 2006 to 2008, he was a Researcher at IT Lab, Fujitsu R&D Center. His team won the first place of both "Text Localization in Real Scenes" and "Text Localization in Born-Digital Images" in the ICDAR 2013 Robust Reading Competition. His current research interests include machine learning, information retrieval, and document analysis and recognition. He is a member of IEEE.

Xuwang Yin received the B.Sc degree in computer science from the University of Science and Technology Beijing, Beijing, China, in 2011. Currently, he is a Graduate Student with the Department of Computer Science and Technology, University of Science and Technology Beijing. His current research interests include machine learning, information retrieval, and document analysis and recognition.



Kaizhu Huang received the Ph.D. degree from the Chinese University of Hong Kong, Hong Kong, in 2004. He is currently an Associate Professor with the Department of Electrical and Electronic Engineering, Xi'an Jiaotong-Liverpool University, Jiangsu, China. In 2008 and 2009, he was respectively a Research Fellow with the Chinese University of Hong Kong and a Post-Doc Researcher at the University of Bristol, Bristol, U.K. From 2009 to 2012, he was an Associate Professor at the Institute of Automation, Chinese Academy of Sciences, China. His current research interests include machine learning and data mining. He is the recipient of APNNA Young Researcher Award 2011. He has published over 80 research papers (JMLR, Neural Computation, NIPS, IJCAI, UAI, ICML, and CVPR).

Hong-Wei Hao received the Ph.D. degree in 1997 from the Institute of Automation, Chinese Academy of Sciences, China, where he is currently a Professor. From 1999 to 2011, he was an Associate Professor and then a Professor at the University of Science and Technology Beijing, Beijing, China. From 2002 to 2003, he was a Visiting Researcher with the Central Research Laboratory, Hitachi Ltd., Tokyo, Japan. His current research interests include large-scale semantic computing theory and technology, large-scale machine learning theory, and intelligent massive information processing.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/publications/dlib.