Robust Text Detection in Natural Scene Images

Xu-Cheng Yin, 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 threshold of the clustering algorithm 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 an character classifier; text candidates with high probabilities are then eliminated and finally texts are identified with a text classifier. The proposed system is evaluated on the ICDAR 2011 Robust Reading Competition dataset; the f measure is over 76% and is significantly better than the state-of-the-art performance of 71%. Experimental results on a publicly available multilingual dataset also show that our proposed method can outperform the other competitive method with the f measure increase of over 9 percent. Finally, we have setup an online demo of our proposed scene text detection system at *"http://kems.ustb.edu.cn/learning/yin/dtext"*.

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, mobile based text analysis and recognition [**?**], [**?**], [**?**], [**?**], [**?**]. Due to complex background, variations of font, size, color and orientation, text in natural scene images has to be robustly detected before being recognized or retrieved.

Existing methods for scene text detection can roughly be categorized into three groups: sliding window based methods [**?**], [**?**], [**?**], connected component based methods [**?**], [**?**], [**?**], and hybrid methods [**?**]. Sliding window based methods, also called as region based methods, engage a sliding window to search for possible texts in the image and then use machine learning techniques to identify texts. These methods tend to be 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. [**?**] 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 (CRFs) [**?**] model, and characters can finally be grouped into text. More recently, Maximally Stable Extremal Region (MSER) based methods, which actually fall into the family of connected component based methods but use MSERs [**?**] as character candidates, have become the focus of several recent projects [**?**], [**?**], [**?**], [**?**], [**?**], [**?**], [**?**].

Although the MSER based method is the winning method of the benchmark data, i.e., ICDAR 2011 Robust Reading Competition [**?**] and has reported promising performance, there remains several problems to be addressed. First, as the MSER algorithm detects a large number of non-characters, most of the character candidates need to be removed before further processing. The existing methods for MSERs pruning [**?**], [**?**], on one hand, may still have room for further improvement in terms of the accuracies; on the other hand, they tend to be slow because of the computation of complex features. Second, current approaches [**?**], [**?**], [**?**] 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 hand-tuned parameters, which is time consuming and error pruning; the clustering based method [**?**] shows good performance but it is complicated by incorporating a second stage processing 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 designed a fast and accurate MSERs pruning algorithm; the number of character candidates to be processed is significantly reduced with a high accuracy. Second, we propose a novel self-training distance metric learning algorithm that can learn distance weights and

- *X.-C. 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: 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.*
- *K. Huang is with Department of Electrical and Electronic Engineering, Xi'an Jiaotong-Liverpool University, Suzhou 215123, China.*
- *H.-W. Hao is with Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China.*

clustering threshold simultaneously and 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 probabilities. Such elimination helps to train a more powerful text classifier for identifying text. By integrating the above ideas, we built an accurate and robust scene text detection system. The system is evaluated on the benchmark ICDAR 2011 Robust Reading Competition dataset and achieved an f measure of 76%. To our best knowledge, this result ranks the first on all the reported performance and is much higher the current best performance of 71%. We also validate our method on the multilingual (include Chinese and English) dataset used in [**?**]. With an f measure of 74.58%, our system significantly outperforms the competitive method [**?**] that achieves only 65.2%. An online demo of our proposed scene text detection system is available at *http://kems.ustb.edu.cn/learning/yin/dtext*.

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 the experimental results of the proposed system on ICDAR 2011 Robust Reading Competition dataset and a multilingual (include Chinese and English) dataset. 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 large number of non-characters candidates in detection and also insufficient text candidates construction algorithms. In this section, we review the MSER based methods with the focus on these two problems. Other scene text detection methods can be referred to in some survey papers [**?**], [**?**], [**?**]. A recently published MSER based method can be referred to in Shi et al. [**?**].

The main advantage of MSER based methods over traditional connected component based methods may root in the usage of MSERs as character candidates. Although the MSER algorithm can detect most characters even when the image is in low quality (low resolution, strong noises, low contrast, etc.), most of the detected character candidates correspond to non-characters. Carlos et al. [**?**] presented a MSERs pruning algorithm that contains two steps: (1) reduction of linear segments and (2) hierarchical filtering. The first stage reduces linear segments in the MSER tree into one node by maximizing the *border energy* function; the second stage walks through the tree in a depth-first manner and eliminates nodes by checking them against a cascade of filters: *size, aspect ratio, complexity, border energy and texture*. Neumann and Matas [**?**] presented a two stage algorithm for Extremal Regions (ERs) pruning. 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 $p(r|\text{chracter})$ 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. As most of the MSERs correspond to non-characters, the purpose of using cascading filters and incrementally computable descriptors in these above two methods is to deal with the computational complexity caused by the high false positive rate.

Another challenge of MSER based methods, or more generally, CC-based methods and hybrid methods, is how to group character candidates into text candidates. The existing methods for text candidates construction fall into two general approaches: rule-based [**?**], [**?**], [**?**] and clustering-based methods [**?**]. Neumann and Matas [**?**] grouped character candidates using the text line constrains, whose basic assumption is that characters in a word can be fitted by one or more top and bottom lines. Carlos et al. [**?**] constructed a fully connected graph over character candidates; they 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. [**?**] pairwised character candidates as clusters by putting constrains on stroke width and height difference; they then exploited a straight line to fit to the centroids of clusters and declared a line as text candidate if it connected three or more character candidates. The clustering-based method presented by Pan et al. [**?**] clusters character candidates into a tree using the minimum spanning tree algorithm with a learned distance metric [**?**]; 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 the energy model.

3 Robust Scene Text Detection

In this paper, by incorporating several key improvements over traditional MSER based methods, we propose a novel MSER based scene text detection method, which finally leads to significant performance improvement over the other competitive methods. The structure of the proposed system, as well as the sample result of each stage is presented in Figure 1. The proposed scene text detection method includes the following stages:

1) *Character candidates extraction*. character candidates are extracted using the MSER algorithm; most of the non-characters are reduced by the proposed MSERs pruning algorithm using the strategy of minimizing

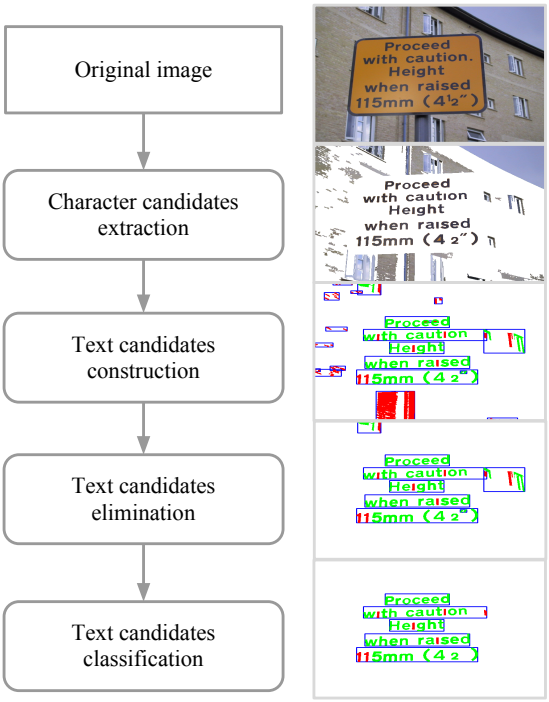


Fig. 1: Flowchart of the proposed system and the corresponding experimental results after each step of a sample image. Text candidates are labeled by blue bounding rectangles; character candidates identified as characters are colored green, others red.

regularized variations. More details are presented in Section 3.1.

2) *Text candidates construction*. distance weights and 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-text are measured using the character classifier and text candidates with high probabilities for non-text are removed. More details are presented in Section 3.3.

4) *Text candidates classification*. text candidates corresponding to true text are identified by the text classifier. An AdaBoost classifier is trained to decide whether an text candidate corresponding to true text or not [**?**]. As characters in the same text tend to have similar features, the uniformity of character candidates' features are used as text candidate's features to train the classifier.

In order to measure the performance of the proposed system using the ICDAR 2011 competition dataset, text candidates identified as text are further partitioned into words by classifying inner character distances into character spacings and word spacings using an AdaBoost classifier [**?**]. The following features are adopted: spacing aspect ratio, relative width difference between left and right neighbors, number of character candidates in the text candidate.

3.1 Letter Candidates Extraction

3.1.1 Pruning Algorithm Overview

The MSER algorithm is able to detect almost all characters even when the image is in low quality. However, as shown in Figure 3a, most of the detected character candidates correspond to non-characters and should be removed before further processing. Figure 3a also shows that the detected characters forms a tree, which is quite useful for designing the pruning algorithm. In real world situations, as characters cannot be "included" by or "include" other characters, it is safe to remove children once the parent is known to be a character, and vice versa. The parent-children elimination is a safe operation because characters are preserved after the operation. By reduction, if the MSER tree is pruned by applying parent-children elimination operation recursively in a depth-first manner, we are still in safe place and characters are preserved. As shown in Figure 3e, the above algorithm will end up with a set of disconnected nodes containing all the characters. The problem with the above algorithm is that it is expensive to identify character. Fortunately, rather than identifying the character, the choice between parent and children can be made by simply choosing the one that is more likely to be characters, which can be estimated by the proposed regularized variation scheme. Considering different situations in MSER trees, we design two versions of the parent-children elimination method, namely the *linear reduction* and *tree accumulation* algorithm. Non-character regions are eliminated by the linear reduction and tree accumulation algorithm using the strategy of minimizing regularized variations. Our experiment on ICDAR 2011 competition training set shows that more than 80% of character candidates are eliminated using the proposed pruning algorithm.

In the following sections, we first introduce the concept of variation and explain why variations need to be regularized. Then we introduce the linear reduction and tree accumulation algorithm. Finally we present the complexity analysis for the proposed algorithms.

3.1.2 Variation and Its Regularization

According to Matas et al. [**?**], 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 are extracted by applying a set of increasing intensity levels to the gray scale image. When the intensity level increases, a new extremal region is extracted by accumulating pixels of current level and joining lower level extremal regions [**?**]; when the top level is reached, extremal regions of the whole image are extracted as a rooted tree. The variation of an extremal region is defined as follows. Let R_l be an extremal region, $B(R_l) = (R_l, R_{l+1}, ..., R_{l+\Delta})$ (Δ is an parameter) be the branch of the tree rooted at R_l, the variation (instability) of R_l is defined as

$$v(R_l) = \frac{|R_{l+\Delta} - R_l|}{|R_l|}. \quad (1)$$

An extremal region R_l is a maximally stable extremal region if its variation is lower than (more stable) its parent R_{l-1} and child R_{l+1} [**?**], [**?**]. Informally, a maximally stable extremal region is an extremal region whose size remains virtually unchanged over a range of intensity levels [**?**].

As MSERs with lower variations have sharper borders and are more likely to be characters, one possible strategy may be used by 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 Figure 2. The children of the MSER tree in Figure 2a correspond to characters while the parent of the MSRE tree in Figure 2b corresponds to character. The "minimize variation" strategy cannot deal with this situation because either parent or children may have the lowest variations. However, our experiment shows that this limitation can be easily fixed by variation regularization, whose 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, 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

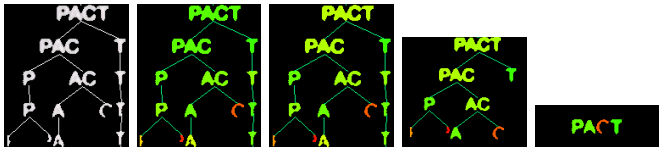
$$\mathcal{V} = \begin{cases} \mathcal{V} - \theta_1(a - a_{max}) & \text{if } a > a_{max} \\ \mathcal{V} - \theta_2(a_{min} - a) & \text{if } a < a_{min} \\ \mathcal{V} & \text{otherwise} \end{cases}, \tag{2}$$

where θ_1 and θ_2 are penalty parameters. Based on experiments on the training dataset, these parameters are set as $\theta_1 = 0.03, \theta_2 = 0.08, a_{max} = 1.2$ and $a_{min} = 0.7$.

(a) (b)

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

Figure 3b shows a MSER 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 Figure 3c. The MSER tree in Figure 3c are used in our linear reduction (result presented in Figure 3d) and tree accumulation algorithm (result presented in Figure 3e). Notice that "variation" in the following sections refer to "regularized variation".



(a) (b) (c) (d) (e)

Fig. 3: MSERs pruning. (a) MSER 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) MSER tree after linear reduction; (e) character candidates after tree accumulation.

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 Figure 4. Given a MSER 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, returns 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 with c, link c's children to t and return t; otherwise we 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. Figure 3d shows the resulting MSER tree after applying linear reduction to the tree shown in Figure 3c. Note that in the resulting tree all linear segments are reduced and non-leaf nodes always have more than one children.

```
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: The linear reduction algorithm.

3.1.4 Tree Accumulation

The tree accumulation algorithm is used when MSERs has more than one child. Given a MSER tree, the procedure returns a set of disconnected nodes. The 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, else discard t's children and return t. Figure 3e shows the result of applying tree accumulation to the tree shown in Figure 3d. Note that the final result is a set of disconnected nodes containing all the characters in the original MSER tree.

```
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: The tree accumulation algorithm.

3.1.5 Complexity Analysis

The linear reduction and tree accumulation algorithm effectively visit each nodes in the MSRE tree and do simple comparisons and pointer manipulations, thus the complexity is linear to the number of tree nodes. The computational complexity of the variation regularization is mostly due to the calculations of MSERs' bounding rectangles, which is up-bounded by the number of pixels in the image.

3.2 Text Candidates Construction

3.2.1 Text Candidates Construction Algorithm Overview

Text candidates are constructed by clustering character candidates using the single-link clustering algorithm [**?**]. Intuitively, single-link clustering produce clusters that are elongated [**?**] 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 progress 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 the above algorithm, each data point represent a character candidate and *top level* clusters in the final hierarchical cluster tree (forest) correspond to text candidates.

The problem is of course 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, 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 distance threshold, can be learned using the proposed distance metric learning algorithm.

In the following subsections, we first introduce the feature space $x_{u,v}$, then detail the proposed metric learning algorithm and finally present the empirical analysis on the proposed 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 top left corner of u's bounding rectangle, h_u, w_u be the height and width of the bounding rectangle of u, s_u be the stroke width of u, $c1_u, c2_u, c3_u$ be the average three channel color value of u, feature vector $x_{u,v}$ include the following features:

- Spatial distance

$$abs(x_u + 0.5h_u - x_v - 0.5w_u)/\max(w_u, w_v).$$

- 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

$$\arctan(\frac{abs(y_u - y_v)}{abs(x_u + 0.5h_u - x_v - 0.5w_u)}),$$
$$\arctan(\frac{abs(y_u + h_u - y_v - h_v)}{abs(x_u + 0.5h_u - x_v - 0.5w_u)}).$$

- Color difference

$$\sqrt{(c1_u - c1_v)^2 + (c2_u - c2_v)^2 + (c3_u - c3_v)^2}.$$

- Stroke width difference

$$abs(s_u - s_v)/\max(s_u, s_v).$$

3.2.3 Distance Metric Learning

There are a variety of distance metric learning methods [**?**], [**?**], [**?**]. More specifically, 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 [**?**], [**?**], [**?**]. The strategy of metric learning is to the learn 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, because clusters are formed by merging smaller clusters, the final resulting clusters will form a binary hierarchical 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 each top level cluster' subclusters are less or equal to ϵ and distances between data pairs in different top level clusters are great than ϵ, in which the distance between clusters is that of the two 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

$$\mathcal{C} = \{(\hat{u}_k, \hat{v}_k) = \underset{u \in C_k, v \in C_{-k}}{\arg\min}\ d(u, v; w)\}_{k=1}^m, \tag{4}$$

$$\mathcal{M} = \{(u_k^*, v_k^*) = \underset{u \in C_k^1, v \in C_k^2}{\arg\min}\ d(u, v; w)\}_{k=1}^m, \tag{5}$$

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. As we know, the logistic regression loss is the traditional loss used in classification with a high and stable performance. By adopting the objective function of logistic regression, we define the following objective function

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

where

$$h_\theta(x'_{u,v}) = 1/(1 + \exp(-\theta^T x'_{u,v})), \tag{9}$$

$$\theta = \begin{pmatrix} -\epsilon \\ w \end{pmatrix},$$

$$x'_{u,v} = \begin{pmatrix} 1 \\ x_{u,v} \end{pmatrix}.$$

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

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

Minimization of the above objective function is a typical nonlinear optimization problem and can be solved by classic gradient optimization methods [**?**].

Note that in the above learning scheme, initial values for w have to be specified in order to generate set $\mathcal{C}$ and $\mathcal{M}$ according to Equation (4) and (5). For this reason, we design an iterative optimization algorithm in which each iteration involves two successive steps corresponding to assignments of $\mathcal{C}, \mathcal{M}$ and optimization with respect to $\mathcal{C}, \mathcal{M}$. We call our algorithm as "*self-training distance metric learning*". Pseudocode for this learning algorithm is presented in Figure 6. Given the top level cluster set $\{C_k\}_{k=1}^m$, the learning algorithm find 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, initial value for θ is set before the iteration begins; in the first stage of the iteration $\mathcal{M}$ and $\mathcal{C}$ are update 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.

Input: labeled clusters set $\{C_k\}_{k=1}^m$
Output: optimized θ such that objective function J is minimized
Method:
randomly initialize θ
repeat
  stage1: update $\mathcal{M}$ and $\mathcal{C}$ with respect to θ
  stage2: $\theta \leftarrow \arg\min_\theta J(\theta : \mathcal{C}, \mathcal{M})$
until convergence or reach iteration limitation

Fig. 6: The self-training distance metric learning algorithm.

Similar to most self-training algorithms, convergence of the proposed algorithm is not guaranteed because the objective function is not assured to decrease in stage one. However, self-training algorithms have demonstrated their success in many applications. In our case, we find that our algorithm can usually generate very good performance after a very small number of iterations,

typically in 5 iterations. This phenomenon will be investigated in the next subsection.

3.2.4 Empirical Analysis

We perform an empirical analysis on the proposed distance metric learning algorithm. We labeled in the ICDAR 2011 competition dataset 466 text candidates corresponding to true text in the training set, 70% of which used as training data, 30% as validation data. In each iteration of the algorithm, cannot-link set $\mathcal{C}$ and must-link set $\mathcal{M}$ are updated in step one by generating cannot-link point pairs and must-link point pairs from true text candidates in every image in the training dataset; the objective function are optimized using the L-BFGS method [**?**] and the parameters are updated in stage two. Performance of the learned distance weights and threshold in step two is evaluated on the validation dataset in each iteration.

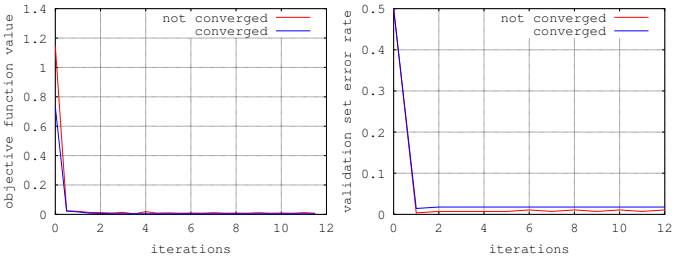
As discussed in the previous section, the algorithm may or may not converge due to different initial values of the parameters. Our experiments show that the learned parameters almost always have a very low error rate on the validation set after the first several iterations and no major improvement is observed in the continuing iterations. As a result, whether the algorithm converge or not has no great impact on the performance of the learned parameters.

We plot the value of the objective function after stage one and stage two in each iteration of two instance (correspond to a converged one and not converged one) of the algorithm in Figure 7a. The corresponding error rates of the learned parameters on the validation set in each iteration are plotted in Figure 7b. Notice that value of the objective function and validation set error rate dropped immediately after the first several iterations. Figure 7b shows that the learned parameters have different error rates due to different initial value, 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 competition training set shows that only 9% of the text candidates correspond to true text. As it is hard to train an effective text classifier using such unbalanced dataset, 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 probabilities for non-text.

The following features are used to train the character classifier. Smoothness, defined as the average difference of adjacent boundary pixels' gradient directions, stroke width features, including the average stroke width and



(a) (b)

Fig. 7: Objective function value (a) and validation set error rate of learned parameters (b) after stage one and stage two in each iteration of two instance of the metric learning algorithm; the red line corresponds to the not converged instance and the blue line corresponds to the converged instance.

stroke width variation, height, width, and aspect ratio. Characters with small aspect ratios such as "i", "j" and "l" are treated 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, of which n ($n \in \mathbb{N}, n \leq m$) are classified as non-characters by a character classifier of precision p ($0 < p < 1$). The probability of the observation conditioning on T corresponding 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

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

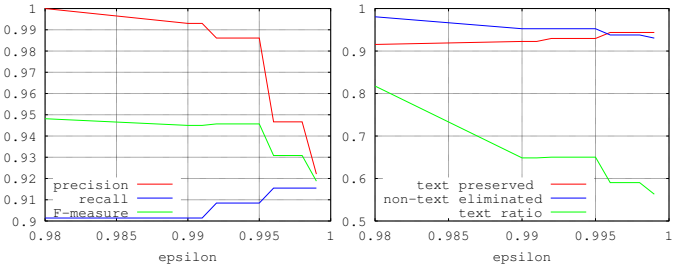
The candidate region is rejected if

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

where ε is the threshold.

Our experiment shows that text candidates of different sizes tend to have different probability of being text. For example, on the ICDAR training set, 1.25% of text candiates of size two correspond to text, while 30.67% of text candidates of size seven correspond to text, which suggests to set different priors for text candidates of different size. Given a text candidates T of size s, let N_s be the total number of text candidates of size s, N_s^* be the number of text candidates of size 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 $P_s(\text{non-text}) = 1 - P_s(\text{text})$. These priors are computed based on statistics on the ICDAR training dataset.



(a) (b)

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.

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

4 Experimental Results

[1]

In this section, we presented the experimental results of the proposed scene text detection method on two publicly available benchmark datasets, ICDAR 2011 Robust Reading Competition dataset [2] and the multilingual dataset [3] provided by Pan et al. [**?**].

4.1 Experiments on ICDAR 2011 Competition Dataset

The ICDAR 2011 Robust Reading Competition (Challenge 2: Reading Text in Scene Images) dataset [**?**] is a widely used dataset for benchmarking scene text detection algorithms. The dataset contains 229 training images and 255 testing images. The proposed system is trained on the training set and evaluated on the testing set.

1. An online demo of the proposed scene text detection system is available at *http://kems.ustb.edu.cn/learning/yin/dtext*.

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

3. The multilingual dataset is available at *http://liama.ia.ac.cn/wiki/projects:pal:member:yfpan*.

It is worth noting that the evaluation scheme of ICDAR 2011 competition is not the same as of ICDAR 2003 and ICDAR 2005. The new scheme, the *object count/area* scheme proposed by Wolf et al. [**?**], is more complicated but offers several enhancements over the old scheme. Basically, these two scheme use the notation of precision, recall and f measure that is defined as

$$recall = \frac{\sum_{i=1}^{|G|} match_G(G_i)}{|G|}, \tag{14}$$

$$precision = \frac{\sum_{j=1}^{|D|} match_D(D_j)}{|D|}, \tag{15}$$

$$f = 2\frac{recall \cdot precision}{recall + precision}, \tag{16}$$

where G is the set of groundtruth rectangles and D is the set of detected rectangles. In the old evaluation scheme, the matching functions are defined as

$$match_G(G_i) = \max_{j=1\ldots|D|} \frac{2 \cdot area(G_i \cap D_j)}{area(G_i) + area(D_j)}, \tag{17}$$

$$match_D(D_j) = \max_{i=1\ldots|G|} \frac{2 \cdot area(D_j \cap G_i)}{area(D_j) + area(G_i)}. \tag{18}$$

The above matching functions only consider one-to-one matches between groundtruth and detected rectangles, leaving room for ambiguity between detection quantity and quality [**?**]. In the new evaluation scheme, the matching functions are redesigned considering detection quality and different matching situations (one-to-one matchings, one-to-many matchings and many-to-one matchings) between groundtruth rectangles and detected rectangles, such that the detection quantity and quality can both be observed using the new evaluation scheme. The evaluation software DetEval [4] used by ICDAR 2011 competition is available online and free to use.

The performance of our system, together with Neumann and Matas' method [**?**], a very recent MSER based method by Shi et al. [**?**] and some of the top scoring methods (Kim's method, Yi's method, TH-TextLoc system and Neumann's method) from ICDAR 2011 Competition are presented in Table 1. As can be seen from Table 1, our method produced much better recall, precision and f measure over other methods on this dataset. It is worth noting that the first four methods in Table 1 are all MSER based methods and Kim's method is the winning method of ICDAR 2011 Robust Reading Competition. 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 on a Linux laptop with Intel (R) Core (TM)2 Duo 2.00GHZ CPU is 0.43s per image. The processing speed of Shi et al.'s method [**?**] on a PC with Intel (R) Core (TM)2 Duo 2.33GHZ CPU is 1.5s per image. The average processing speed of Neumann and Matas' method [**?**] is

4. DetEval is available at *http://liris.cnrs.fr/christian.wolf/software/deteval/index.html*.

1.8s per image on a "standard PC". Figure 9 shows some text detection examples by our system on ICDAR 2011 dataset.

Fig. 9: Text detection examples on the ICDAR 2011 dataset. Detected text by our system are labeled using red rectangles. Notice the robustness against low contrast, complex background and font variations.

TABLE 1: Performance (%) comparison of text detection algorithms on ICDAR 2011 Robust Reading Competition dataset.

Methods	Recall	Precision	f
Our Method	**68.26**	**86.29**	**76.22**
Shi et al.'s method [**?**]	63.1	83.3	71.8
Kim's Method (not published)	62.47	82.98	71.28
Neumann and Matas [**?**]	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

To fully appreciate the benefits of *text candidates elimination* and *the MSERs pruning algorithm*, we further profiled the proposed system on this dataset using the following schemes (see Table 2)

1) **Scheme-I**, no text candidates elimination performed. As can be seen from Table 2, the absence of text candidates elimination results in a major decrease in precision value. The degradation can be explained by the fact that large number of non-text are passed to the text candidates classification stage without being eliminated.

2) **Scheme-II**, using default parameter setting [**?**] for the MSER extraction algorithm. The MSER 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. As can be seen from Table 2, compared with our parameter setting ($\Delta = 1, v_+ = 0.5, d_+ = 0.1$), the default parameter setting ($\Delta = 5, v_+ = 0.25, d_+ = 0.2$) results in a major decrease in recall value. The degradation can be explained by two reasons: (1) the MSER algorithm is not able to detect some low contrast characters (due to v_+), and (2) the MSER algorithm tends to miss some regions that are more likely to be characters (due to Δ and d_+). Note that the speed loss (from 0.36 seconds to 0.43 seconds) is mostly due to the MSER detection algorithm itself.

TABLE 2: Performance (%) of the proposed method due to different components

Component	Recall	Precision	f	Speed (s)
Overall system	68.26	86.29	76.22	0.43
Scheme-I	65.57	77.49	71.03	0.41
Scheme-II	61.63	85.78	71.72	0.36

4.2 Experiments on Multilingual Dataset

The multilingual (include Chinese and English, see Figure 10) dataset was initially published by Pan et al. [**?**] to evaluate the performance of their scene text detection system. The training dataset contains 248 images and the testing dateset contains 239 images. As there are no apparent spacing between Chinese word, this multilingual dataset only provides groundtruths for text lines. We hence evaluate the text line detection performance of our system without further partitioning text into words. Figure 10 shows some scene text detection examples by our system on this dataset.

TABLE 3: Performance (%) comparison of text detection algorithms on the multilingual dataset. Speed of Pan et al.'s method is profiled on a PC with Pentium D 3.4GHz CPU.

Methods	Recall	Precision	f	Speed (s)
Scheme-III	63.23	79.38	70.39	0.22
Scheme-IV	68.45	82.63	74.58	0.22
Pan et al.'s method [**?**]	65.9	64.5	65.2	3.11

Fig. 10: Text detection examples on the multilingual dataset. Detected text by our system are labeled using red rectangles.

The performance of our system (include **Scheme-III** and **Scheme-IV**) and Pan et al.'s method [**?**] is presented in Table 3. The evaluation scheme in ICDAR 2003 competition (see Section 4.1) is used for fair comparison. The main difference between **Scheme-III** and **Scheme-IV** is that the character classifier in the first scheme

is trained on the ICDAR 2011 training set while the character classifier in the second scheme is trained on the multilingual training set (character features for training the classifier are the same). The result comparison between **Scheme-III** and **Scheme-IV** in Table 3 shows that the performance of the proposed system is significantly improved because of the incorporating of the Chinese-friendly character classifier. The basic implication of this improvement is that the character classifier has a significant impact on the performance of the overall system, which offers another advantage of the proposed system: the character classifier can be trained on desired dataset until it is accurate enough and be plugged into the system and the overall performance will be improved. Table 3 also shows the advantages of the proposed method over Pan et al.'s method in detection quality and speed.

5 Conclusion

This paper presents a new MSER based scene text detection method. Several key improvement over traditional methods have been proposed. We propose a fast and accurate MSERs pruning algorithm that enables us to detect most the characters even when the image is in low quality. We propose a novel self-training distance metric learning algorithm that can learn distance weights and threshold simultaneously; text candidates are constructed by clustering character candidates by the single-link algorithm using the learned parameters. We propose to use a character classifier to estimate the posterior probability of text candidate corresponding to non-text and eliminate text candidates with high probability for non-text, which helps to build a more powerful text classifier. By integrating the above ideas, we built a robust scene text detection system that exhibited superior performance over state-of-the-art methods on both the ICDAR 2011 Competition dataset and a multilingual dataset.

Acknowledgments

The research was partly supported by National Basic Research Program of China (2012CB316301) and National Natural Science Foundation of China (61105018, 61175020).