A Unified Framework for Multi-Oriented Text Detection and Recognition

Cong Yao, Xiang Bai, *Member, IEEE*, and Wenyu Liu, *Member, IEEE*

***Abstract*—High level semantics embodied in scene texts are both rich and clear and thus can serve as important cues for a wide range of vision applications, for instance, image understanding, image indexing, video search, geolocation and automatic navigation. In this paper, we present a unified framework for text detection and recognition in natural images. The contributions of this work are three-fold: (1) Text detection and recognition are accomplished concurrently using exactly the *same* features and classification scheme. (2) In contrast to methods in the literature, which mainly focus on horizontal or near-horizontal texts, the proposed system is capable of localizing and reading texts of varying orientations. (3) A new dictionary search method is proposed, to correct the recognition errors usually caused by confusions among similar yet different characters. As an additional contribution, a novel image database with texts of different scales, colors, fonts and orientations in diverse real-world scenarios, is generated and released. Extensive experiments on standard benchmarks as well as the proposed database demonstrate that the proposed system achieves highly competitive performance, especially on multi-oriented texts.**

***Index Terms*—Unified framework, text detection, text recognition, natural image, arbitrary orientations, dictionary search.**

I. Introduction

Unlike conventional low level visual cues such as interest points, contours and regions, which are usually ambiguous and unreliable, texts in natural scenes directly carry rich and clear high level semantics and thus can provide useful information for a wide range of vision tasks, for instance, image understanding [1], image search [2], target geolocation [3], and robot navigation [4]. This property has made text detection and recognition in natural images active research topics in computer vision [5]–[19]. Since scene text images usually contain a large amount of irrelevant elements in addition to useful text contents, text detection is a critical procedure to localize texts and discard irrelevant elements. Then text recognition is required to interpret the symbols in the localized text regions and to convert them into computer readable and editable characters, thus making accessible the high level semantics embodied in texts.

Fig. 1. Detection and recognition examples by the proposed system. The system can detect and recognize texts of different scales, colors, fonts and orientations, in various real-world scenarios. Notably, even curved, longitudinally aligned, and reversed words can be successfully localized and read.

Previous methods either only tackle one of these two tasks [5], [9], [13], [18], [20] or treat them as two isolated stages in an end-to-end text recognition system [6], [17], [21]. In this paper, we propose a unified framework, which takes text detection and recognition as a whole and performs both tasks in a single unified pipeline.

As stated in [22], in conventional end-to-end scene text recognition systems [10], [21], decisions of different components in the pipeline are sequential and isolated, which makes it impossible to exploit feedback information between different components. In this work, we investigate this issue by sharing the features for text detection and recognition and make joint decisions for these two tasks. Feature sharing, which leverages common features shared across different object classes to train classifiers jointly, has proven to be very effective in multi-class object detection [23], [24] and recognition [25]. In this paper, we extend this concept to the case where tasks at different levels are involved. Text detection (classification between text and non-text) is a coarse level task, while its consequent task, character recognition (discrimination among different character classes) is at a fine level. Shared features are used for both tasks in the proposed framework. Specifically, we perform text detection and character recognition using exactly the same features (see Sec. III-C for details).

Most of the existing methods for text detection [6], [10], [11], [26] and recognition [7], [8], [18] can only handle horizontal or near-horizontal texts. This is a severe drawback, as a large portion of the texts in real-world scenarios are non-horizontal. Such limitation would make it fail to capture the information embodied in non-horizontal texts and thus seriously restricts the practicability of these methods. To remedy this issue, we propose an algorithm that can detect and recognize

Copyright (c) 2014 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending a request to pubs-permissions@ieee.org.

This work was primarily supported by National Natural Science Foundation of China (NSFC) (No. 61222308), and in part by NSFC (No. 61173120 and 60903096), Program for New Century Excellent Talents in University (No. NCET-12-0217) and Fundamental Research Funds for the Central Universities (No. HUST 2013TS115). The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Aleksandra Pizurica. *(Corresponding author: Xiang Bai.)*

The authors are all with the Department of Electronics and Information Engineering, Huazhong University of Science and Technology (HUST), Wuhan, 430074, China (e-mail: yaocong2010@gmail.com; xbai@hust.edu.cn; liuwy@hust.edu.cn).

texts with different orientations. Meanwhile, the proposed algorithm is robust to variations in scale, color and font. Fig. 1 depicts several typical end-to-end text recognition examples by the proposed system. As can be seen, the algorithm is able to detect and recognize texts of different scales, colors, fonts and orientations, in various real-world scenarios. Notably, even curved, longitudinally aligned, and reversed words can be successfully localized and read.

In order to correct errors in character recognition, language prior and context information (e.g. n-gram statistics [20], [27]–[29], lexicon or dictionary [18], [21], [30], [31]) are usually employed. In general, for scene text recognition dictionary based correction methods are more effective than n-gram based methods as the latter require sufficient context information in text, which is unavailable in scene images as there are far less texts compared to document images. Moreover, dictionary based correction methods are able to handle partial missing and spurious characters, which often occur for scene text. In this paper, we propose a modified dictionary search method, based on the Levenshtein edit distance [32], to correct errors in recognition (see Sec. III-E).

To assess the effectiveness of the proposed framework, we have conducted extensive experiments on text detection and end-to-end text recognition using public benchmarks, the ICDAR 2011 dataset [33], the Chars74K dataset *et al.* [9], and the MSRA-TD500 dataset [34]. Images from the MSRA-TD500 dataset are primarily for benchmarking text detection algorithms and contain both English and Chinese characters, but most text recognition methods (including the work presented in this paper) have focused on English letters and Arabic numbers. So we also generate a large collection of natural images with only English letters and Arabic numbers, for the purpose of evaluating end-to-end text recognition systems. The details of this database can be found in Sec. IV-A. The proposed algorithm achieves state-of-the-art or highly competitive performance on ICDAR 2011, Chars74K, and MSRA-TD500, as well as on the novel database.

In summary, the contributions of the paper are as follows: (1) We propose a unified framework, which takes scene text detection and recognition as a whole, in contrast to other methods that treat detection and recognition as two sequential and isolated stages. (2) We show that text detection and recognition can be accomplished concurrently with exactly the same features and classification scheme. (3) This is the first work that is capable of detecting and recognizing texts of arbitrary orientations in complex natural scenes. (4) A novel dictionary search method for recognition error correction is proposed. (5) A large and challenging image database containing texts with high variability in complex natural scenes is generated and made publicly available.

In our previous work [34], we proposed a detection system for texts of arbitrary orientations. This paper is a continuation and extension of [34]. The entire architecture, including the features, classifiers, and algorithm pipeline, is inherited from [34]. However, certain key modifications are made to build an end-to-end system for both text detection and recognition. Specifically, the classification scheme is modified to accomplish classification between text and non-text as well as discrimination among different character classes; a dictionary search based error correction method for text recognition is introduced; and solutions for the issues in multi-oriented text recognition and case disambiguation are presented. See Sec. III for details.

The remainder of this article is organized as follows. Section II reviews previous methods in the literature related to the work presented in this paper. Section III describes the main ideas and details of the unified framework for scene text detection and recognition. The experiments on both text detection and recognition as well as discussions are presented in Section IV. Finally, we conclude the paper and point out potential directions for future research in Section V.

II. Related Work

Driven by the wide range of potential applications, scene text detection and recognition have been important research topics for decades [7], [8], [35]–[41] and have recently seen a surge in research efforts in the computer vision community [9]–[14], [17], [18], [20], [21], [26], [42]. Comprehensive surveys on text detection and recognition in static images and videos can be found in [43]–[45]. This literature review will concentrates upon end-to-end scene text recognition systems.

TextFinder proposed by Wu *et al.* [39] was a representative system for automatic text detection and recognition in the early years. This system treated text as a distinctive texture and procedures called texture segmentation and chip generation were applied in succession to extract candidate regions. These candidate regions were then refined to recover missed strokes and remove false positives. Finally, the detected text regions were binarized and fed to a commercial OCR software for character recognition.

The approaches of Chen *et al.* [6] and Epshtein *et al.* [10] are recent attempts towards end-to-end text recognition systems. These approaches chiefly address the problem of text detection and achieve subsequent recognition using off-the-shelf OCR engines. In [6], Chen *et al.* trained a cascade classifier to select text regions and applied commercial OCR software to the detected regions. Epshtein *et al.* [10] proposed a novel image operator, called Stroke Width Transform (SWT), to localize texts in natural images. For recognition, off-the-shelf OCR engine was adopted to recognize the masks produced by SWT. These methods have achieved excellent performance on natural images and inspired numerous researchers in this field [42], [46], [47]. However, these methods treat text detection and recognition as isolated stages and only focus on horizontal texts. The recognition accuracy of these methods is fairly limited as off-the-shelf OCR engines are specifically designed for texts in document images, not natural images.

Off-the-shelf OCR software was also used in the text detection and recognition pipeline proposed in [7]. Different from the algorithms of Chen *et al.* [6] and Epshtein *et al.* [10], which only apply OCR once for each text region, this method utilized a multiple hypotheses recognition scheme to cope with errors in binarization. The text image is segmented and recognized several times and the final result is selected from the generated text string hypotheses based on their confidences. The main

drawback of this method is the low efficiency resulting from its repeated segmentation and recognition operations. In addition, this method is not applicable to non-horizontal texts.

In [12], Neumann and Matas presented a Maximally Stable Extremal Region (MSER) [48] based method for text localization and recognition, which obtained impressive detection and recognition results on challenging real-world images. The authors achieved scene text recognition using a trained classifier based on synthetic fonts instead of off-the-shelf OCR software. Moreover, to achieve higher performance this algorithm replaced the traditional feed-forward pipeline with a hypotheses-verification framework that kept multiple hypotheses at each stage and allowed feedback loops between different stages for hypothesis verification. Similar to most of the methods, this algorithm only handles horizontal texts.

Wang *et al.* [21] introduced an end-to-end scene text recognition system, which aimed to tackle a special case of the scene text understanding problem where in addition to the natural image it is also given a list of words (i.e., a lexicon) to be detected and read. However, the usability of this algorithm in general text understanding scenarios is limited since a lexicon with probable words for each individual image is not always available. Recently, Neumann and Matas [17] propose a real-time[1] scene text localization and recognition method. This algorithm first generates character candidates using Extremal Regions (instead of Maximally Stable Extremal Regions (MSERs) [48]), then selects suitable Extremal Regions by a sequential classifier and groups the selected Extremal Regions into words, and finally recognizes them using a trained classifier. This method is robust to noise and low contrast. This algorithm treats text detection and recognition as separate stages and is only applicable to horizontal texts.

Most related to our work, Weinman *et al.* [22] suggested to tightly couples several aspects of text detection and recognition and share features for these two tasks. In this paper, we achieve text detection and recognition concurrently with exactly the same features and classification scheme. In this sense, we truly realize a unified framework for scene text detection and recognition.

The algorithm in our previous work [34] is able to detect scene texts with various orientations and the features are effective as well for recognizing characters. Therefore, we base current work on the method of [34] to build an end-to-end scene text recognition system, which is simple yet effective and is capable of localizing and reading texts of different orientations.

III. Methodology

In this section, we present the main ideas and details of the proposed algorithm. Specifically, we give an overview of the proposed framework in Sec. III-A, review the previous work on multi-oriented text detection in Sec. III-B, explain the classification scheme for simultaneous text detection and character recognition in Sec. III-C, illustrate the new dictionary search method for error correction in Sec. III-E, and describe the details of the training data in Sec. III-G.

[1]A text recognition system is considered real-time if the processing time is comparable with the time to read the text for a human, as stated in [17].

A. Overview

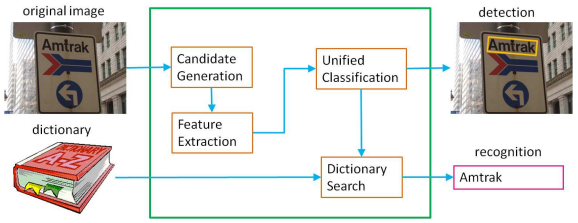

Fig. 2. Schematic overview of the proposed framework.

A schematic overview of the proposed framework is illustrated in Fig. 2. Given a natural image, candidates (characters and lines) are firstly generated using SWT [10] and clustering. In contrast to conventional methods which use separate features and classifiers for text detection and recognition, we perform these two tasks simultaneously using the same features and classification scheme. The initial recognition results may be erroneous and thus are fed to a dictionary-based correction module to correct the errors. The final outcomes of the framework are detected text regions and recognized characters.

B. Previous Work on Multi-Oriented Text Detection

In our previous work [34], two sets of features, component level features and chain level features, and two classifiers, component level classifier and chain level classifier, are adopted to construct a system for multi-oriented text detection in natural images.

In the pipeline, pixels are first grouped into connected components using SWT [10], corresponding to strokes or characters; connected components are then linked to chains by clustering, corresponding to words or sentences. The connected components and chains are verified by the informative features and discriminative classifiers.

The component level features and chain level features, which capture the intrinsic properties of text and are not specific to scale and rotation, are designed to describe components and chains and discriminate between text and non-text. The component level classifier and chain level classifier are both Random Forest [49] trained on the component level features and chain level features, respectively. For more details of this algorithm, please refer to [34].

C. Classification Scheme for Text Detection and Character Recognition

It has been a time-honoured tradition in the computer vision community to use trees (or randomized trees) as indexing structure for various tasks, for example, shape recognition [50], keypoint recognition [51], image classification [52], image segmentation [53] and human pose detection [54]. Enabling scalability and allowing efficient nearest neighbour search are the main reasons for choosing trees (or randomized trees). Following this tradition, we also adopt randomized trees to index samples for character recognition.

According to preliminary experiments, the component level features in [34], which are originally designed for text detection, are also excellent character descriptors that can distinguish among different character classes. Therefore, we make use of this property to construct a classification scheme for both text detection and recognition.

The whole architecture of the system in [34], including the features, classifiers, and algorithm pipeline, is kept intact, except for the component level classifier. The component level classifier (Random Forest) is modified to perform both text detection and recognition, i.e. simultaneously discriminating between text and non-text components and distinguishing among different character classes.

Briefly speaking, each leaf node of the component level classifier is associated with an extra histogram which stores the empirical distribution of character classes, such that each leaf node becomes a weak classifier that can predict the probability of an example to be true text and the probability of the example to be a specific character class. This modification will be described in detail below.

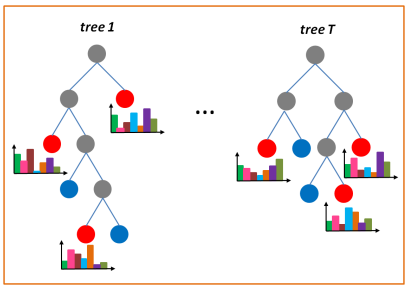

Fig. 3. Illustration of character distribution histograms. Since the trees are exhaustively grown, each leaf node is either positive (red) or negative (blue). For each positive leaf node, a character distribution histogram is computed using the examples falling into it and stored for future use.

As illustrated in Fig 3, in the training procedure for the component level classifier, at each positive leaf node[2] of the tree in the forest an histogram is computed and stored for future use. This histogram is computed on the basis of the examples that fall in the associated leaf node.

Each training example x_i is associated with two labels, a visible label y_i^v and a hidden label y_i^h. y_i^v indicates whether x_i is text component or not. y_i^h is the character index if x_i is text component and the invalid label if not.

$$y_i^v = \begin{cases} 1 & \text{if } x_i \text{ is text component} \\ 0 & \text{otherwise} \end{cases}, \tag{1}$$

$$y_i^h = \begin{cases} \alpha_i & \text{if } y_i^v = 1 \\ * & \text{otherwise} \end{cases}, \tag{2}$$

where $*$ stands for invalid label, $\alpha_i \in \Omega$ is the character index of x_i and Ω is the alphabet. In this paper, we consider English letters (52 classes) and Arabic numbers (10 classes), i.e. $\Omega = \{a, \ldots, z; A, \ldots, Z; 0, \ldots, 9\}$ and $|\Omega| = 62$.

When growing the trees only the visible labels are used to split the nodes. The character distribution histogram of each positive leaf node is computed using the hidden labels (corresponding to character classes) of the training examples falling into it. For positive leaf node l, the character distribution histogram is:

$$h_l(\alpha) = \frac{\sum_{k=1}^{n_l} \mathbb{1}(y_k^h = \alpha)}{n_l}, \alpha \in \Omega, \tag{3}$$

where n_l is the number of the examples in l and $\mathbb{1}(\cdot)$ is the indicator function.

In the testing procedure, the component level classifier makes two predictions for an unseen component x, one is the probability of x to be a text component, $p(x)$, and the other is the probability of x to be a specific character α, $q_x(\alpha)$. $p(x)$ is a real value and $q_x(\alpha)$ is a histogram where the value of each bin represents the probability of x to be the corresponding class. x will finally reach a leaf node (which could be either positive or negative) in each of the T trees. Let $\{l_{x,t}\}_{t=1}^T$ denote these leaf nodes, $p(x)$ and $q_x(\alpha)$ can be computed as follows:

$$p(x) = \frac{\sum_{t=1}^T \varphi(l_{x,t})}{T}, \tag{4}$$

$$q_x(\alpha) = \frac{\sum_{t=1}^T \hbar_{l_{x,t}}(\alpha)}{T}, \tag{5}$$

where $\varphi(\cdot)$ is a function to indicate whether a leaf node is positive:

$$\varphi(l) = \begin{cases} 1 & \text{if } l \text{ is positive} \\ 0 & \text{otherwise} \end{cases}. \tag{6}$$

$\hbar_{l_{x,t}}(\alpha)$ is the response of leaf node $l_{x,t}$ to predict which character class x is likely to be and it is defined as:

$$\hbar_{l_{x,t}}(\alpha) = \begin{cases} h_{l_{x,t}}(\alpha) & \text{if } \varphi(l_{x,t}) = 1 \\ 1/|\Omega| & \text{otherwise} \end{cases}. \tag{7}$$

Since no character distribution histogram is stored for negative leaf nodes, uniform distribution is expected if x reaches a negative leaf node.

In the way described above, the component level classifier is able to achieve text detection and recognition at the same time. Its effectiveness relies on two properties: (1) The discriminative power of the component level features, which can not only classify text components from non-text components but also distinguish among different characters. (2) The inherent clustering mechanism of random trees, i.e. examples falling into the same leaf node tend to be similar in some aspects [52].

Extensive experiments on text detection, character recognition and end-to-end text recognition confirm the effectiveness of this extension to the original randomized trees. See the experiments in Sec. IV for details. Though we only demonstrate this classification scheme on scene text detection and recognition, it is quite general and applicable to other problems, in which tasks of different levels are involved and features can be shared by these tasks.

D. Component Linking and Word Partition

Upon investigation, we found that the accuracy of text detection depends heavily on the component linking and word partition process. Since text recognition takes as input the consequence of text detection, the quality of component linking and word partition also has significant impact on the accuracy

[2]A leaf node is positive if all the examples in this node are positive.

of text recognition. In this work, we adopt the strategies for candidate linking and word partition proposed in [55], to replace those in our previous work [34], leading to excellent text detection and recognition performance (see Sec. IV). The pipelines for component linking and word partition are from the method in [55], but we used a similar yet slightly different set of features to measure the similarities and spacings among components, since we have estimated a group of rotation-invariant properties of components (such as center, scale and major orientation).

E. Error Correction in Character Recognition

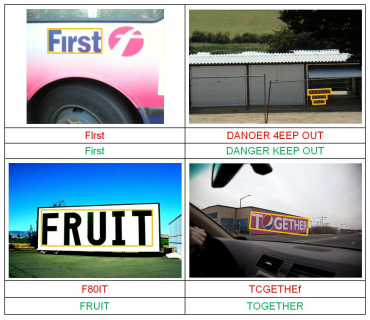

Fig. 4. Necessity of error correction in character recognition. The red strings ($\xi(\omega_i)$) are raw outputs, which are obtained by simply picking the top choice for each character while the green strings ($\eta(\omega_i)$) are recognition results after error correction.

Obviously, for a detected string, simply picking for each character the top choice predicted by the classifier, i.e. the character class with the highest classification score, would lead to poor character recognition [28]. Therefore, as shown in Fig. 4, error correction is necessary to improve the final character recognition performance. In this work we propose a new dictionary based search method to correct recognition errors.

1) Dictionary: The dictionary used in this work is built from the top 100k frequently searched words on the Bing search engine [56]. The word list is provided by the Microsoft Web N-Gram Service [57].

There are mainly two reasons for choosing a list of frequently searched words instead of a traditional dictionary: (1) The word list includes the words that are of real interest to people all over the world. For example, the list contains many names of people and places, which are usually not included in a traditional dictionary. (2) In contrast to dictionaries, the words in the list are ranked according to the search frequency, which can provide useful information for dictionary search.

This dictionary is assumed to be universal, thus it is used in all the experiments on all the datasets.

2) Dictionary Search: As texts of arbitrary orientations are considered, the arrangement of a text line is not necessarily from left to right, instead it might be from top to bottom, or even from right to left, as shown in Fig. 1. We assume text is a type of linear sequence and words within a text line are in the same order. Thus for a given text line, there are two possible arrangement orders: One is in accord with the linking order of the characters and the other is contrary to the linking order. We denote these two arrangements as normal order and reverse order, respectively. The arrangement of a text line is inferred from its property.

All the detected texts are divided into separate words using the method as described in [10]. For each word ω_i, the raw recognition $\xi(\omega_i)$ is a string obtained by picking the top choice for each character $\omega_{i,j}$ in this word. Both $\xi(\omega_i)$ and its reverse version $\xi^{\leftarrow}(\omega_i)$ are matched with all the items in the dictionary. The two items with the largest similarity with $\xi(\omega_i)$ and $\xi^{\leftarrow}(\omega_i)$, denoted by $\eta(\omega_i)$ and $\eta^{\leftarrow}(\omega_i)$, as well as the word similarities $s(\omega_i)$ and $s^{\leftarrow}(\omega_i)$, are recorded. The definition of similarity s will be described in Sec. III-E3.

To cope with out-of-dictionary words and numbers, a threshold τ is introduced. If the value of the similarity $s(\omega_i)$ and $s^{\leftarrow}(\omega_i)$ are both lower than τ, $\eta(\omega_i)$ and $\eta^{\leftarrow}(\omega_i)$ are replaced by strings computed by applying n-gram based correction [28] to $\xi(\omega_i)$ and $\xi^{\leftarrow}(\omega_i)$). $s(\omega_i)$ and $s^{\leftarrow}(\omega_i)$ are set to a constant. The threshold τ is empirically set to 0.8 for all the experiments in this paper.

For a text line L with N words $\omega_i, i = 1, 2, ...N$, total similarity of normal order and that of reverse order are defined as:

$$S(L) = \sum_{i=1}^{N} s(\omega_i), \tag{8}$$

$$S^{\leftarrow}(L) = \sum_{i=1}^{N} s^{\leftarrow}(\omega_i). \tag{9}$$

The order of L, $O(L)$, is then determined as follows:

$$O(L) = \begin{cases} \oplus & \text{if } S(L) \geqq S^{\leftarrow}(L) \\ \ominus & \text{otherwise} \end{cases}, \tag{10}$$

where $\oplus$ stands for normal order and $\ominus$ stands for reverse order.

If L is in normal order, the final recognition result is the concatenation of all the $\eta(\omega_i), i = 1, 2, ...N$; otherwise, the final recognition result is the concatenation of all the $\eta^{\leftarrow}(\omega_i), i = 1, 2, ...N$.

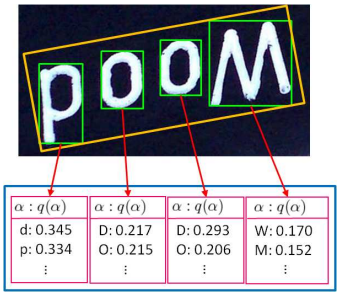

Fig. 5. Probabilities of character classes (only top choices are shown). The word in the image is "Wood". Certain characters can be very confusing. For example, after rotation the letter 'd' is very similar to 'p'.

3) Definition of Similarity: In the original Levenshtein edit distance [32], three basic operations, deletion, insertion and substitution, are allowable. In string matching, the costs of the three operations are all defined as a constant 1. However, it is not appropriate to define constant cost for substitution, because certain characters can be very confusing, while certain characters are distinctive from each other. Therefore, replacements between different character pairs should have different

costs, depending on the similarity between them. For example, in the image in Fig. 5, replacing the first character 'd' with 'z' should receive more penalty than replacing it with 'p', since the difference between $q(\alpha = d)$ and $q(\alpha = p)$ is marginal.

For a given character x, we redefine the cost of substituting θ with ϑ in an adaptive manner:

$$c_{subs}(\vartheta \Leftarrow \theta|x) = 1-\log_2(1+\min(\frac{q_x(\alpha=\vartheta)}{q_x(\alpha=\theta)}, \frac{q_x(\alpha=\theta)}{q_x(\alpha=\vartheta)})). \quad (11)$$

$q_x(\alpha)$ is a histogram where the value of each bin represents the probability of x to be the corresponding class. According to this definition, if θ and ϑ are very similar, the substitution cost is close to zero; conversely, if θ and ϑ are totally dissimilar, the substitution cost is approximately equal to 1.

As stated in Sec. III-E1 the rank of dictionary items is also helpful. Intuitively, if two words in the dictionary have proximal distances to the query, the one ranked higher should be selected with priority. Therefore, we define the similarity between a query word and a dictionary item using the rank of the dictionary item in addition to their edit distance.

Assume that the normalized edit distance [58] between a query word ω and a dictionary item ϖ is $d(\omega, \varpi)$ and the relative rank of ϖ in the dictionary is $r(\varpi)$. The similarity between ω and ϖ is:

$$s(\omega, \varpi) = \lambda \cdot \frac{1}{1 + d(\omega, \varpi)} + (1 - \lambda) \cdot \frac{1}{1 + r(\varpi)}, \quad (12)$$

where $\lambda \in [0, 1]$ is a control parameter. The optimal value of λ is determined through experiment (see Sec. IV-D).

F. Case Disambiguation

In some applications (e.g. book digitalization), case sensitive character recognition is required. However, the case of letters cannot be solely determined by the classifier, as there exists inherent case ambiguity for some letters, for instance, 's' and 'S'. Case disambiguation is therefore a necessary step to dispel the case ambiguity.

Following the work of Novikova *et al.* [18], we consider three types of word layout: (1) All letters are in upper case. (2) All letters are in lower case. (3) The first letter is in upper case and the rest in lower case. Note that it is unnecessary to perform case disambiguation for numbers.

We use a case disambiguation method similar to that of Novikova *et al.* [18]. The mean score of all letters except for the first one is computed to estimate the case of the tail of the word. If all the letters in the tail are in upper case, the first letter is also taken as upper case. Otherwise we estimate the case of the first letter based on the relative size between the first letter and the rest letters. Instead of specifically training SVM classifiers for distinguishing upper case English letter from the same lower case one, we directly utilize the prediction of the component level classifier.

G. Training Data

It is extremely labor-intensive and time-consuming to collect and label a large collection of text images (positive examples) and background images (negative examples). However, in previous works [9], [12], [21], it has been proven effective in character detection and recognition to use synthetic fonts in place of real-world characters. Following these works, we automatically extract positive examples from synthetic text images and negative examples from natural images. Two naive classifiers (a component level classifier and a chain level classifier) are trained using a subset of the training images, to bootstrap the training examples. Sample images from the training set are shown in Fig. 6.

For positive examples, we synthesize 100k images using the program[3] of Wang *et al.* [21]. Each synthetic image contains a string of 2 to 12 random characters (letter or digit). 200 fonts installed in the Windows operating system are used to synthesize the text images. The bounding boxes of the characters are generated and recorded automatically by the program. To make the synthetic images more realistic, random affine transformation, Gaussian noise and blur are applied to each synthetic image. Different from previous works [12], [21], the synthetic texts in this paper may be with varying orientations, not restricted to horizontal or near-horizontal. See the images on the left of Fig. 6.

We use random strings instead of real words in dictionary to synthesize the text images, as real words would introduce a priori bias to the training data. For example, in English letter 'E' and 'A' occur frequently while 'Z' and 'J' are very rare. This priori bias may give rise to errors in character recognition.

For negative examples, we select 30k natural images that don't contain any text from six public datasets: Berkeley Segmentation Data Set and Benchmarks 500 (BSDS500)[4] [59], Zurich Building Image Database[5] [60], Oxford Buildings Dataset[6] [61], MIT-CBCL StreetScenes Dataset[7] [62], CASIA Tampered Image Detection Evaluation Database (CAISA TIDE) V2.0[8] [63], and PASCAL VOC 2011 Dataset[9] [64].

It is noted that a large portion of false positives in text detection are caused by man-made stuff (e.g. bricks and windows) and vegetation (e.g. grasses and leaves), as these elements may exhibit a regular structure similar to text. To make the system more robust to these distractors, we selected a tremendous amount of images with various street views and outdoor scenes, in which man-made stuff and vegetation are very common (See the images on the right of Fig. 6).

IV. Experiments

We have trained a unique model using the training data described in Sec. III-G. 600 trees are used for training the component level classifier and 300 trees for the chain level classifier. About 2M component level examples and 560k chain level examples are bootstrapped from the training data for learning this model.

[3] http://vision.ucsd.edu/∼kai/grocr/

[4] http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/resources.html

[5] http://www.vision.ee.ethz.ch/showroom/zubud/

[6] http://www.robots.ox.ac.uk/∼vgg/data/oxbuildings/

[7] http://cbcl.mit.edu/software-datasets/streetscenes/

[8] Credits for the use of the CASIA Image Tempering Detection Evaluation Database (CAISA TIDE) V2.0 are given to the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Science, Corel Image Database and the photographers. http://forensics.idealtest.org

[9] http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2011/

Fig. 6. Examples of training data. *Left*: Synthetic images (positive). *Right*: Natural images (negative).

We apply this unique model with the same parameter setting to all the datasets evaluated in this paper. We do not derive models specific to each dataset as the goal of this work is to build a general system for scene text detection and recognition.

A. Datasets

ICDAR 2011 The ICDAR 2011 dataset [33] is an extension to the dataset used for the text locating competitions of ICDAR 2003 [65] and 2005 [66]. The ICDAR 2011 dataset includes 485 natural images in total. All the images in this database are from the dataset of ICDAR 2003 and 2005, except for a few extra images added by the organizers of the ICDAR 2011 Robust Reading Competition Challenge 2[10]. The ground truth rectangles have been relabelled due to the problems with the dataset used in the previous ICDAR competitions (for example, imprecise bounding boxes and inconsistent definitions of "word"). Moreover, the previous evaluation protocol is replaced by the evaluation method proposed by Wolf *et al.* [67], as the latter is able to handle the cases of one-to-many and many-to-many matches. We performed both text detection and recognition on this dataset.

Chars74K The Chars74K dataset[11] was released by de Campos *et al.* [9] to evaluate recognition algorithms for individual characters in natural images. This dataset contains symbols of both English and Kannada language and only part of the characters in the original images are annotated. Therefore we assess our system on the *GoodImg* dataset, a subset of the Chars74K dataset, following Neumann *et al.* [12]. The *GoodImg* dataset includes 636 images, in which a portion of Latin letters and Arabic numbers are labelled by the annotators. Character recognition performances instead of word recognition performances are reported for this dataset, because only annotations for individual characters are available.

MSRA-TD500 The MSRA-TD500 dataset[12] is a benchmark for evaluating detection algorithms for multi-oriented texts in natural scenes, which was first introduced in our early work [34]. This dataset consists of 500 images with horizontal as well as slant and skewed texts in complex natural scenes. An evaluation protocol was also devised for this dataset, since traditional evaluation methods are primarily designed for horizontal texts. We only assess the detection ability of our system on this database, excluding the images with Chinese characters, which are not supported by our current implementation.

HUST-TR400 Since there is no standard benchmark for multi-oriented scene text recognition, we collect a large database of images containing English letters and Arabic numbers of different fonts, sizes, colors and orientations, to better assess the capability of the proposed algorithm.

This database is diverse in both text and background. The images are from three sources: (1) Images taken by the volunteers for this project. These images are shot in various cities (New York, Philadelphia, San Diego, Providence, etc.) in America by several people using different devices (iPhone, pocket camera and single lens reflex camera). (2) Images harvested from the Flickr website[13]. These images are searched using key words like "street view", "billboard", "guide board", "highway", "house number", "book cover", "shop" and "store". (3) Images from the MSRA-TD500 dataset. We adopt some images that only contain English letters and Arabic numbers from MSRA-TD500 to augment this database. This database is generated for the purpose of benchmarking end-to-end scene text recognition systems and includes 400 images in all, so we name it HUST Text Recognition 400 Database (HUST-TR400)[14].

Some typical images of this dataset are shown in Fig. 7. In contrast to the Chars74K dataset, all the images in HUST-TR400 are fully annotated at word level, making it possible to evaluate and compare word recognition performances of different scene text recognition methods. HUST-TR400 is very challenging as it contains images with diverse texts (different fonts, colors, sizes and orientations) in various real-world scenarios (street, highway, restaurant, office, mall and so on). Due to its diversity, HUST-TR400 can serve as a standard benchmark for end-to-end scene text recognition.

The Street View Text (SVT) dataset proposed by Wang *et al.* [21] is also a famous benchmark in this field. However, this database only provides incomplete annotations for words and the task of [21] is to localize and recognize words listed in an image-specific lexicon, which differs significantly from the general task of scene text recognition. The NEOCR dataset [68] includes images with multi-oriented texts in natural scenes. However, the texts in this database are in different languages other than English, such as Hungarian, Russian, Turkish and Czech, which are not supported by our end-to-end recognition system. Hence, we didn't assess our

[10]http://robustreading.opendfki.de/wiki/SceneText
[11]http://www.ee.surrey.ac.uk/CVSSP/demos/chars74k/
[12]http://www.loni.ucla.edu/∼ztu/publication/MSRA-TD500.zip
[13]http://www.flickr.com/
[14]Will be available at http://mc.eistar.net/

Fig. 7. Typical images of the HUST Text Recognition 400 Database (HUST-TR400).

TABLE I
PERFORMANCES OF DIFFERENT TEXT DETECTION METHODS EVALUATED ON THE ICDAR 2011 DATASET [33].

Algorithm	Precision	Recall	F-measure
Proposed	0.822	0.657	0.730
Yin *et al.* [55], [69]	**0.863**	0.683	**0.762**
Neumann *et al.* [70]	0.854	0.675	0.754
Koo *et al.* [71]	0.814	**0.687**	0.745
Shi *et al.* [72]	0.833	0.631	0.718
Kim *et al.* [33]	0.830	0.625	0.713
Neumann *et al.* [17]	0.731	0.647	0.687
Yi *et al.* [73]	0.672	0.581	0.623
Yang *et al.* [33]	0.670	0.577	0.620
Neumann *et al.* [33]	0.689	0.525	0.596
Shao *et al.* [33]	0.635	0.535	0.581

TABLE II
PERFORMANCES OF DIFFERENT TEXT DETECTION METHODS EVALUATED ON THE MSRA-TD500 DATASET [34].

Algorithm	Precision	Recall	F-measure
Proposed	0.64	0.62	0.61
Yin *et al.* [55]	**0.71**	0.61	**0.66**
TD-Mixture [34]	0.63	**0.63**	0.60
TD-ICDAR [34]	0.53	0.52	0.50
Epshtein *et al.* [10]	0.25	0.25	0.25
Chen *et al.* [6]	0.05	0.05	0.05

algorithm on these two datasets.

B. Experimental Results and Discussions

1) Detection Results on Horizontal Texts: We tested the proposed system on the ICDAR 2011 dataset [33], which is the newest benchmark for the famous ICDAR Robust Reading competition. The quantitative detection performance as well as that of other methods on this benchmark are shown in Tab. I. The results for comparison are quoted from the original papers or the summary of the ICDAR Robust Reading competition [33]. The proposed system achieves promising performance (precision=0.822, recall=0.657 and F-measure=0.730) on this dataset, but still behind the top performing algorithms [55], [69]–[71]. This is reasonable since the proposed algorithm is primarily designed for multi-oriented texts, while those in [55], [69]–[71] are mostly focused on horizontal texts.

2) Detection Results on Multi-Oriented Texts: To further assess the detection functionality of the proposed system, we applied it to the MSRA-TD500 dataset [34], in which texts in the images may be with different directions, fonts, colors and scales. Tab. II shows the performances of different methods on this database. The proposed method performs well on this dataset, outperforming TD-Mixture [34] in F-measure, but with inferior performance than the state-of-the-art algorithm [55]. Note that the training data for the proposed method is independent of the training set of MSRA-TD500.

Fig. 8. Examples of scene text detection and recognition on the Chars74K dataset [9]. Incorrect recognition results are marked in red.

3) Individual Character Recognition Results on Multi-Oriented Texts: Texts in the images from the Chars74K dataset [9] may be with different orientations. This makes the dataset suitable for assessing the proposed system, because the algorithm is capable of detecting and recognizing texts of varying orientations. Fig. 8 depicts some text detection and recognition examples of the proposed algorithm on this database.

To enable fair comparison, we follow the evaluation protocol of Neumann *et al.* [12]. For each labelled character in the *GoodImg* subset, three situations are considered: (1) *Matched*. The character is correctly localized and recognized. (2) *Mismatched*. The character is localized correctly but recognized incorrectly. (3) *Not found*. The character is missed in the detection phase. It is impossible to assess word level recognition performance on this dataset, as individual characters rather than full words are annotated for this database.

The quantitative results of the proposed algorithm and Neumann *et al.* [12] are presented in Tab. III. The proposed

TABLE III
INDIVIDUAL CHARACTER RECOGNITION RESULTS OF DIFFERENT METHODS ON THE CHARS74K DATASET [9].

Algorithm	Matched	Mismatched	Not found
Proposed	75.9%	11.3%	12.8%
Neumann *et al.* [12]	71.6%	12.1%	16.3%
de Campos *et al.* [9]	54.3%	45.7%	N/A

algorithm achieves better performance than the method of Neumann *et al.* [12]. Specifically, the proposed algorithm obtains lower miss rate and higher recognition accuracy. There are mainly two possible reasons: (1) The method of Neumann *et al.* is primarily designed for horizontal or near-horizontal texts, while our algorithm is able to localize and recognize texts of different orientations (see Fig. 8). (2) We adopted more advanced strategy for component linking [55], which further boosts the performance of text detection and recognition.

The result of de Campos *et al.* [9] is also included in Tab. III. The method of de Campos *et al.* assumed perfect detection and used manually located characters for recognition. In contrast, both the proposed system and that of Neumann *et al.* [12] detect the characters automatically.

There are also many Kannada letters in the images of the Chars74k dataset. These Kannada letters can be successfully detected but not correctly read by the proposed system (see Fig. 8), since the current implementation does not support the Kannada alphabet. However, these detected Kannada letters didn't affect the quantitative results and comparison, as the algorithms were purely evaluated on English ground truth.

Fig. 9. Examples of scene text detection and recognition on horizontal texts.

4) End-to-End Recognition Results on Horizontal Texts: We also tested the proposed unified framework on the ICDAR 2011 dataset [33] to evaluate its ability of extracting text information from natural scenes. Several end-to-end scene text detection and recognition examples are shown in Fig. 9. The proposed system is able to localize and read texts of different fonts, colors and sizes, in complex natural scenes.

The proposed method was compared against the systems of Neumann *et al.* [17] and Milyaev *et al.* [74]. We used the same evaluation method as Neumann *et al.* [17]. A word is regarded as correctly recognized if it is localized with recall higher than 80% and all the letters belonging to the word are recognized correctly with case sensitive comparison. Note that this evaluation protocol is very stringent, as it judges an algorithm purely from the final recognition output. All types of errors in the pipeline, including misses and false positives in detection, imprecise localization, improper word partition, incorrect character recognition, and mistakes in case determination, would lead to failure.

TABLE IV
END-TO-END RECOGNITION PERFORMANCES OF DIFFERENT METHODS EVALUATED ON THE ICDAR 2011 DATASET [33].

Algorithm	Precision	Recall	F-measure
Proposed (case insensitive)	0.528	0.47	0.486
Proposed (case sensitive)	0.492	0.440	0.454
Milyaev *et al.* (case sensitive) [74]	**0.66**	**0.46**	**0.54**
Neumann *et al.* (case sensitive) [75]	0.454	0.448	0.452
Neumann *et al.* (case sensitive) [17]	0.372	0.371	0.365

The quantitative results are shown in Tab. IV. For case sensitive recognition, our system achieves a performance of precision 0.492, recall 0.440 and F-measure 0.454, which slightly outperforms the system of Neumann *et al.* [75] but is inferior than the method of Milyaev *et al.* [74]. For case insensitive recognition, the proposed algorithm achieves even higher performance (precision=0.528, recall=0.473 and F-measure=0.486).

Note that in our algorithm the dictionary used for error correction is independent from the ICDAR 2011 dataset. When incorporating the dictionary[15] for the ICDAR 2011 dataset the proposed algorithm obtains significantly enhanced performance (precision=0.553, recall=0.501 and F-measure=0.512 for case sensitive recognition).

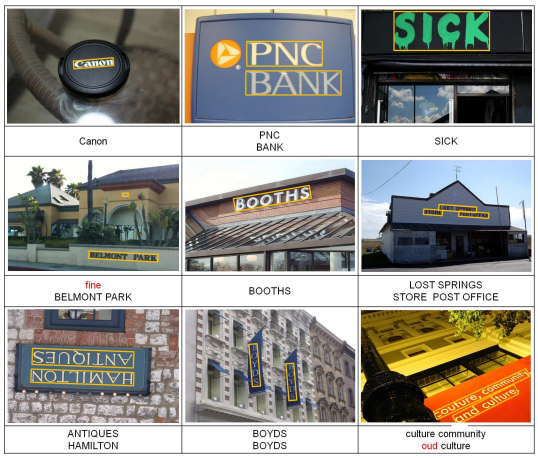

Fig. 10. Examples of scene text detection and recognition on the HUST-TR400 dataset. Incorrect recognition results are marked in red.

5) End-to-End Recognition Results on Multi-Oriented Texts: The HUST-TR400 database includes 400 natural images with texts of high variability in scale, color, font and orientation, in complex real-world scenes. We evaluated the proposed algorithm on this challenging benchmark. All the images in this dataset are for testing. This means one should use extra data to train models for text detection and character recognition. Fig. 10 depicts some end-to-end text recognition examples of the proposed system. As can be seen from Fig. 10, the proposed algorithm works fairly well in diverse real-world scenarios and is robust to variations of texts and background

[15]It is constructed by assembling together all the ground truth words in the test set of the ICDAR 2011 dataset.

TABLE V
END-TO-END RECOGNITION PERFORMANCE OF THE PROPOSED METHOD EVALUATED ON THE HUST-TR400 DATASET.

Algorithm	Precision	Recall	F-measure
Proposed (case insensitive)	0.415	0.386	0.393
Proposed (case sensitive)	0.374	0.344	0.351

TABLE VI
COMPARISON OF DIFFERENT ERROR CORRECTION METHODS EVALUATED ON THE ICDAR 2011 DATASET.

Algorithm	Precision	Recall	F-measure
Traditional	0.477	0.434	0.445
Adaptive subs. cost	0.506	0.455	0.469
Adaptive subs. cost+Relative rank	0.528	0.473	0.486

clutters. In particular, the algorithm is capable of localizing and reading texts of varying directions. Even longitudinally aligned and reversed words can be successfully detected and recognized.

For quantitative assessment, we adopted the evaluation method of Neumann *et al.* [17], which has been described in detail in Sec. IV-B4. Both case sensitive and case insensitive recognition performances of the proposed method on the HUST-TR400 database are shown in Tab. V. There is an improvement of 4.2% in F-measure if case sensitive comparison is replaced by case insensitive comparison when computing recognition accuracy. It is not possible to evaluate performances of competing methods on this dataset, since so far no system for multi-oriented scene text recognition is publicly available.

C. Effectiveness of Proposed Error Correction Method

In this section, we assess the effectiveness of the proposed method for error correction in character recognition. Compared to traditional dictionary search based method, we defined an adaptive substitution cost and incorporated relative rank of words in the dictionary as extra information, in order to improve character recognition accuracy. Therefore, to justify the effectiveness of the proposed error correction method, three settings are considered: (1) *Traditional*. As in traditional methods, both adaptive substitution cost and relative rank of words are not used. (2) *Adaptive substitution cost*. Only adaptive substitution cost is adopted. (3) *Adaptive substitution cost+Relative rank*. Both adaptive substitution cost and relative rank of words are incorporated (proposed method). These settings are evaluated on the ICDAR 2011 dataset [33] and the end-to-end recognition performances (case insensitive) are reported in Tab. VI. As can be seen, the incorporation of adaptive substitution cost results in an improvement of 2.3% in F-measure; relative rank of words can provide additional assistance, leading to a further improvement of 1.7% in F-measure.

D. Impact of Parameter λ

In this section, we investigate the impact of the parameter λ to end-to-end text recognition accuracy. We experimented with different values of λ and computed the end-to-end recognition performance on the ICDAR 2011 dataset [33]. Fig. 11 shows how the F-measure changes when the value of λ varies.

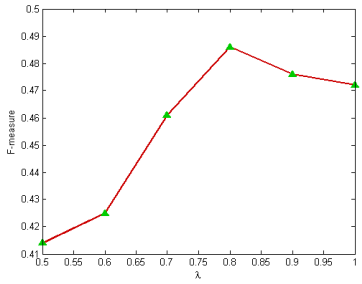

Fig. 11. Impact of Parameter λ.

As can be seen, the highest F-measure is achieved when $\lambda = 0.8$. This indicates edit distance is more important than relative rank in error correction. All the results of the proposed system reported in the previous sections, where applicable, are obtained with $\lambda = 0.8$.

E. Limitations of Proposed Algorithm

Though the proposed algorithm works fairly well under broad real-world scenarios, it is still far from perfect. It would make mistakes in both detection phase and recognition phase under certain conditions.

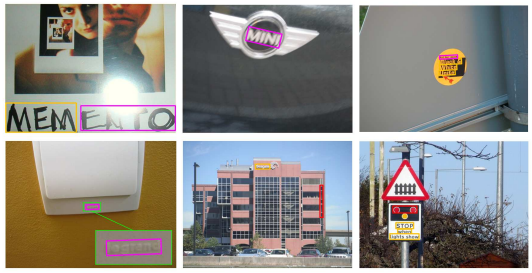

Fig. 12. Typical failure cases of the proposed algorithm in detection. Yellow rectangles: true positives, pink rectangles: false negatives, red rectangles: false positives. Best viewed in color.

Fig. 12 illustrates some typical failure cases of text detection. The misses are mainly due to non-uniform lighting condition, blur, low resolution and low contrast between text and background; the false positives are mostly caused by patterns that are very similar to true text, such as windows and signs.

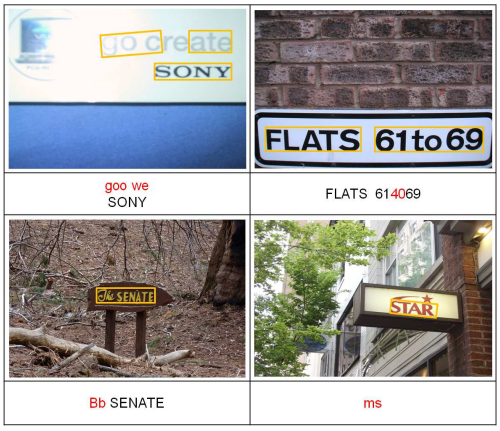

Fig. 13. Typical failure cases of the proposed algorithm in recognition. Incorrect recognition results are marked in red.

Several typical failure cases of character recognition are shown in Fig. 13. Partial misses in detection, improper word partitions, irregular fonts, connected characters, all give rise to recognition errors.

In addition, the processing speed of the proposed system is also a limitation. Without optimization and parallelization, the average processing time of the proposed algorithm on the ICDAR 2011 dataset is about 1s and that on the HUST-TR400 dataset is about 3.5s. We believe the processing efficiency of the proposed algorithm can be significantly enhanced by utilizing optimization and parallelization techniques [76], [77].

In conclusion, there is still room for improvement for both text detection and recognition in natural scenes, as related real-world applications have high requirements for reliable text information extraction [44].

V. Conclusions and Future Work

In this paper, we have presented a unified framework for detection and recognition of multi-oriented scene texts. Text detection and recognition are accomplished concurrently with exactly the same features and classification scheme. A dictionary search based method for recognition error correction is also proposed. The proposed system is capable of detecting and recognizing texts of different scales, colors, fonts and orientations, in diverse real-world scenes. Extensive experiments demonstrate that compared to existing methods in the literature the proposed algorithm achieves state-of-the-art or very competitive performance on various challenging benchmarks as well as on a novel database we propose.

Actually, the modification we have made to the Random Forest classifier for the purpose of simultaneous text classification and character recognition is quite general and thus can be applied to other domains and problems, in which tasks are carried out at different levels and features can be shared by these tasks. This direction is worthy of further exploration.

The proposed algorithm can assist numerous applications that require text information extraction from images or videos, such as video search, target geolocation, and automatic navigation. In the future, we will devote ourselves to the development of such practical systems, based on the proposed algorithm.

Acknowledgment

The authors would like to thank the anonymous reviewers for their valuable suggestions.

References

[1] L. V. Remias, *A Real-Time Image Understanding System for an Autonomous Mobile Robot*. Storming Media, 1996.

[2] S. Tsai, H. Chen, D. Chen, G. Schroth, R. Grzeszczuk, and B. Girod, "Mobile visual search on printed documents using text and low bit-rate features," in *Proc. of ICIP*, 2011.

[3] D. B. Barber, J. D. Redding, T. W. McLain, R. W. Beard, and C. N. Taylor, "Vision-based target geo-location using a fixed-wing miniature air vehicle," *Journal of Intelligent and Robotic Systems*, vol. 47, no. 4, pp. 361–382, 2006.

[4] G. N. DeSouza and A. C. Kak, "Vision for mobile robot navigation: A survey," *IEEE Trans. PAMI*, vol. 24, no. 2, pp. 237–267, 2002.

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

[6] X. Chen and A. Yuille, "Detecting and reading text in natural scenes," in *Proc. of CVPR*, 2004.

[7] D. Chen, J. M. Odobez, and H. Bourlard, "Text detection and recognition in images and video frames," *Pattern Recognition*, vol. 37, no. 3, pp. 595–608, 2004.

[8] M. R. Lyu, J. Song, and M. Cai, "A comprehensive method for multilingual video text detection, localization, and extraction," *IEEE Trans. CSVT*, vol. 15, no. 2, pp. 243–255, 2005.

[9] T. E. de Campos, B. R. Babu, and M. Varma, "Character recognition in natural images," in *Proc. of VISAPP*, 2009.

[10] B. Epshtein, E. Ofek, and Y. Wexler, "Detecting text in natural scenes with stroke width transform," in *Proc. of CVPR*, 2010.

[11] K. Wang and S. Belongie, "Word spotting in the wild," in *Proc. of ECCV*, 2010.

[12] L. Neumann and J. Matas, "A method for text localization and recognition in real-world images," in *Proc. of ACCV*, 2010.

[13] C. Yi and Y. Tian, "Text string detection from natural scenes by structure-based partition and grouping," *IEEE Trans. Image Processing*, vol. 20, no. 9, pp. 2594–2605, 2011.

[14] A. Coates, B. Carpenter, C. Case, S. Satheesh, B. Suresh, T. Wang, and A. Y. Ng, "Text detection and character recognition in scene images with unsupervised feature learning," in *Proc. of ICDAR*, 2011.

[15] K. L. Bouman, G. Abdollahian, M. Boutin, and E. J. Delp, "A low complexity sign detection and text localization method for mobile applications," *IEEE Trans. Multimedia*, vol. 13, no. 5, pp. 922–934, 2011.

[16] X. Liu and W. Wang, "Robustly extracting captions in videos based on stroke-like edges and spatio-temporal analysis," *IEEE Trans. Multimedia*, vol. 14, no. 2, pp. 482–489, 2012.

[17] L. Neumann and J. Matas, "Real-time scene text localization and recognition," in *Proc. of CVPR*, 2012.

[18] T. Novikova, O. Barinova, P. Kohli, and V. Lempitsky, "Large-lexicon attribute-consistent text recognition in natural images," in *Proc. of ECCV*, 2012.

[19] C. Yao, X. Bai, B. Shi, and W. Liu, "Strokelets: A learned multi-scale representation for scene text recognition," in *Proc. of CVPR*, 2014.

[20] A. Mishra, K. Alahari, and C. V. Jawahar, "Top-down and bottom-up cues for scene text recognition," in *Proc. of CVPR*, 2012.

[21] K. Wang, B. Babenko, and S. Belongie, "End-to-end scene text recognition," in *Proc. of ICCV*, 2011.

[22] J. J. Weinman, "Unified detection and recognition for reading text in scene images," Ph.D. dissertation, University of Massachusetts Amherst, Amherst, MA, 2008.

[23] A. Torralba, K. P. Murphy, and W. T. Freeman, "Sharing features: efficient boosting procedures for multiclass object detection," in *Proc. of CVPR*, 2004.

[24] ——, "Sharing visual features for multiclass and multiview object detection," *IEEE Trans. PAMI*, vol. 29, no. 5, pp. 854–869, 2007.

[25] E. Murphy-Chutorian and J. Triesch, "Shared features for scalable appearance-based object recognition," in *Proc. of IEEE Workshop Applications of Computer Vision*, 2005.

[26] A. Ikica and P. Peer, "An improved edge profile based method for text detection in images of natural scenes," in *Proc. of EUROCON*, 2011.

[27] X. Tong and D. A. Evans, "A statistical approach to automatic ocr error correction in context," in *Proc. of Fourth Workshop on Very Large Corpora*, 1996.

[28] R. Smith, "Limits on the application of frequency-based language models to ocr," in *Proc. of ICDAR*, 2011.

[29] Y. Bassil and M. Alwani, "Ocr context-sensitive error correction based on google web 1t 5-gram data set," *American Journal of Scientific Research*, vol. 50, 2012.

[30] K. Kukich, "Techniques for automatically correcting words in text," *ACM Computing Surveys*, vol. 24, no. 4, pp. 377–439, 1992.

[31] R. Haldar and D. Mukhopadhyay, "Levenshtein distance technique in dictionary lookup methods: An improved approach," *CoRR*, vol. abs/1101.1232, 2011.

[32] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," *Soviet Physics Doklady*, vol. 10, no. 8, pp. 707–710, 1966.

[33] A. Shahab, F. Shafait, and A. Dengel, "ICDAR 2011 robust reading competition challenge 2: Reading text in scene images," in *Proc. of ICDAR*, 2011.

[34] C. Yao, X. Bai, W. Liu, Y. Ma, and Z. Tu, "Detecting texts of arbitrary orientations in natural images," in *Proc. of CVPR*, 2012.

[35] R. Lienhart and F. Stuber, "Automatic text recognition in digital videos," University of Mannheim, Tech. Rep., 1995.

[36] Y. Zhong, K. Karu, and A. K. Jain, "Locating text in complex color images," *Pattern Recognition*, vol. 28, no. 10, pp. 1523–1535, 1995.

[37] V. Wu, R. Manmatha, and E. M. Riseman, "Finding text in images," in *Proc. of 2nd ACM Int. Conf. Digital Libraries*, 1997.
[38] A. Jain and B. Yu, "Automatic text location in images and video frames," *Pattern Recognition*, vol. 31, no. 12, pp. 2055–2076, 1998.
[39] V. Wu, R. Manmatha, and E. M. Riseman, "TextFinder: An automatic system to detect and recognize text in images," *IEEE Trans. PAMI*, vol. 21, no. 11, pp. 1224–1229, 1999.
[40] Y. Zhong, H. Zhang, and A. K. Jain, "Automatic caption localization in compressed video," *IEEE Trans. PAMI*, vol. 22, no. 4, pp. 385–392, 2000.
[41] H. P. Li, D. Doermann, and O. Kia, "Automatic text detection and tracking in digital video," *IEEE Trans. Image Processing*, vol. 9, no. 1, pp. 147–156, 2000.
[42] Y. Pan, X. Hou, and C. Liu, "A hybrid approach to detect and localize texts in natural scene images," *IEEE Trans. Image Processing*, vol. 20, no. 3, pp. 800–813, 2011.
[43] D. Chen, J. Luettin, and K. Shearer, "A survey of text detection and recognition in images and videos," IDIAP, Tech. Rep., August 2000.
[44] K. Jung, K. Kim, and A. Jain, "Text information extraction in images and video: a survey," *PR*, vol. 37, no. 5, pp. 977–997, 2004.
[45] J. Liang, D. Doermann, and H. Li, "Camera-based analysis of text and documents: a survey," *IJDAR*, vol. 7, no. 2, pp. 84–104, 2005.
[46] H. Chen, S. S. Tsai, G. Schroth, D. M. Chen, R. Grzeszczuk, and B. Girod, "Robust text detection in natural images with edge-enhanced maximally stable extremal regions," in *Proc. of ICIP*, 2011.
[47] A. Mosleh, N. Bouguila, and A. B. Hamza, "Image text detection using a bandlet-based edge detector and stroke width transform," in *Proc. of BMVC*, 2012.
[48] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide baseline stereo from maximally stable extremal regions," in *Proc. of BMVC*, 2002.
[49] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.
[50] Y. Amit and D. Geman, "Shape quantization and recognition with randomized trees," *Neural Computation*, vol. 9, no. 7, pp. 1545–1588, 1997.
[51] V. Lepetit, P. Lagger, and P. Fua, "Randomized trees for real-time keypoint recognition," in *Proc. of CVPR*, 2005.
[52] F. Moosmann, E. Nowak, and F. Jurie, "Randomized clustering forests for image classification," *IEEE Trans. PAMI*, vol. 30, no. 9, pp. 1632–1646, 2008.
[53] J. Shotton, M. Johnson, and R. Cipolla, "Semantic texton forests for image categorization and segmentation," in *Proc. of CVPR*, 2008.
[54] G. Rogez, J. Rihan, S. Ramalingam, C. Orrite, and P. H. S. Torr, "Randomized trees for human pose detection," in *Proc. of CVPR*, 2008.
[55] X. C. Yin, X. Yin, K. Huang, and H. Hao, "Robust text detection in natural scene images," *IEEE Trans. PAMI*, p. preprint, 2013.
[56] M. Corporation, *Bing Search Engine*, 2012, http://www.bing.com/.
[57] ——, *Microsoft Web N-Gram Service (Public Beta)*, 2010, http://web-ngram.research.microsoft.com/info/.
[58] Y. Li and B. Liu, "A normalized levenshtein distance metric," *IEEE Trans. PAMI*, vol. 29, no. 6, pp. 1091–1095, 2007.
[59] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Trans. PAMI*, vol. 33, no. 5, pp. 898–916, 2011.
[60] H. Shao, T. Svoboda, and L. V. Gool, "ZuBuD–zurich buildings database for image based recognition," Swiss Federal Institude of Technology, Tech. Rep., April 2003.
[61] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching," in *Proc. of CVPR*, 2007.
[62] S. M. Bileschi, "Streetscenes: Towards scene understanding in still images," Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA, 2006.
[63] CASIA, *CASIA Tampered Image Detection Evaluation Database*, 2011, http://forensics.idealtest.org/.
[64] M. Everingham, L. V. Gool, C. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2011 (VOC2011) Results," http://www.pascal-network.org/challenges/VOC/voc2011/workshop/index.html.
[65] S. M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, and R. Young, "ICDAR 2003 robust reading competitions," in *Proc. of ICDAR*, 2003.
[66] S. M. Lucas, "ICDAR 2005 text locating competition results," in *Proc. of ICDAR*, 2005.
[67] C. Wolf and J. M. Jolion, "Object count/area graphs for the evaluation of object detection and segmentation algorithms," *IJDAR*, vol. 8, no. 4, pp. 280–296, 2006.
[68] R. Nagy, A. Dicker, and K. Meyer-Wegener, "NEOCR: A configurable dataset for natural image text recognition," in *CBDAR Workshop at ICDAR*, 2011.
[69] X. C. Yin, X. Yin, K. Huang, and H. Hao, "Accurate and robust text detection: a step-in for text retrieval in natural scene images," in *Proc. of SIGIR*, 2013.
[70] L. Neumann and J. Matas, "On combining multiple segmentations in scene text recognition," in *Proc. of ICDAR*, 2013.
[71] H. Koo and D. H. Kim, "Scene text detection via connected component clustering and nontext filtering," *IEEE Trans. Image Processing*, vol. 22, no. 6, pp. 2296–2305, 2013.
[72] C. Shi, C. Wang, B. Xiao, Y. Zhang, and S. Gao, "Scene text detection using graph model built upon maximally stable extremal regions," *Pattern Recognition Letters*, vol. 34, no. 2, pp. 107–116, 2013.
[73] C. Yi and Y. Tian, "Text detection in natural scene images by stroke gabor words," in *Proc. of ICDAR*, 2011.
[74] S. Milyaev, O. Barinova, , T. Novikova, P. Kohli, and V. Lempitsky, "Image binarization for end-to-end text understanding in natural images," in *Proc. of ICDAR*, 2013.
[75] L. Neumann and J. Matas, "Scene text localization and recognition with oriented stroke detection," in *Proc. of ICCV*, 2013.
[76] X. Wang, Z. Zhang, Y. Ma, X. Bai, W. Liu, and Z. Tu, "Robust subspace discovery via relaxed rank minimization," *Neural computation*, vol. 26, no. 3, pp. 611–635, 2014.
[77] C. Yao, X. Bai, W. Liu, and L. J. Latecki, "Human detection using learned part alphabet and pose dictionary," in *Proc. of ECCV*, 2014.

Cong Yao received the B.S. and Ph.D. degrees, both in Electronics and Information Engineering, from Huazhong University of Science and Technology (HUST), Wuhan, China, in 2008 and 2014, respectively. His research has focused on computer vision and machine learning, particularly in the area of text detection and recognition in natural images.

Xiang Bai received the B.S., M.S., and Ph.D. degrees from Huazhong University of Science and Technology (HUST), Wuhan, China, in 2003, 2005, and 2009, respectively, all in Electronics and Information Engineering. He is currently a Professor with the Department of Electronics and Information Engineering, HUST. He is also the Vice-Director of National Center of Anti-Counterfeiting Technology, HUST. His research interests include object recognition, shape analysis, scene text recognition and intelligent systems.

Wenyu Liu received the B.S. degree in Computer Science from Tsinghua University, Beijing, China, in 1986, and the M.S. and Ph.D. degrees, both in Electronics and Information Engineering, from Huazhong University of Science and Technology (HUST), Wuhan, China, in 1991 and 2001, respectively. He is now a professor and associate dean of the Department of Electronics and Information Engineering, HUST. His current research areas include sensor network, multimedia information processing, and computer vision. He is a member of IEEE System, Man and Cybernetics Society.