

# Scene Text Recognition Using Structure-Guided Character Detection and Linguistic Knowledge

Cun-Zhao Shi, *Student Member, IEEE*, Chun-Heng Wang, *Member, IEEE*,  
Bai-Hua Xiao, *Member, IEEE*, and Song Gao, Jin-Long Hu

**Abstract**—Scene text recognition has inspired great interests from the computer vision community in recent years. In this paper, we propose a novel scene text-recognition method integrating structure-guided character detection and linguistic knowledge. We use part-based tree structure to model each category of characters so as to detect and recognize characters simultaneously. Since the character models make use of both the local appearance and global structure informations, the detection results are more reliable. For word recognition, we combine the detection scores and language model into the posterior probability of character sequence from the Bayesian decision view. The final word-recognition result is obtained by maximizing the character sequence posterior probability via Viterbi algorithm. Experimental results on a range of challenging public data sets (ICDAR 2003, ICDAR 2011, SVT) demonstrate that the proposed method achieves state-of-the-art performance both for character detection and word recognition.

**Index Terms**—Character recognition, cropped word recognition, part-based tree-structured models (TSMs), posterior probability, scene text recognition, word spotting.

## I. INTRODUCTION

WITH the rapid growth of camera-based applications readily available on smart phones and portable devices, understanding the pictures taken by these devices semantically has gained increasing attention from the computer vision community in recent years. Among all the information contained in the image, text, which carries semantic information, could provide valuable cues about the content of the image and thus is very important for human as well as computer to understand the scenes. In an experimental study, Judd *et al.* [1] found that given an image containing text and other objects, viewers tend to fixate on text. This further demonstrates that text recognition is very important for humans to understand the scenes. In fact, text recognition is indispensable for a lot of applications such as automatic sign reading, language translation, navigation, and so on.

Generally speaking, to understand the information carried by text in the image, we need to recognize the text detected



Fig. 1. Some scene text images from ICDAR 2003 [2] and SVT [13]. The characters in these images have different fonts, shadows, distortions, deformations, low resolutions, and occlusions.

from the images. The ICDAR 2003 robust reading challenge [2] published the first data set to highlight the problem of detecting and recognizing scene text. In this benchmark, the organizers identified four subtasks: 1) text localization; 2) robust character recognition; 3) robust word recognition; and 4) robust reading. Since text detection is the premise for recognition, a lot of methods have been proposed to address the problem of detecting and localizing text in scene images [3]–[12] and some have reported promising localization performance. For scene text recognition, some previous methods [4], [8], used off-the-shelf optical character recognition (OCR) for subsequent recognition, whose performance was unsatisfactory. Although many commercial OCR systems work well on scanned documents under controlled environment, they perform poorly on scene text images due to the unsatisfactory binarization results of text images of low resolution, unconstrained lighting, distortion, and complex background. Some recent methods [13]–[19] propose to recognize scene text without binarization. They adopt multiscale sliding window strategy to detect characters and directly extract features from the original image to recognize the characters. The performance of some recently proposed methods [15]–[17] is quite promising. However, due to the unconstrained lighting conditions, various fonts, deformations, occlusions, sometimes low resolution, and complex background of text in natural scene images, the performance of scene text recognition is still unsatisfactory. Fig. 1 shows some examples of scene text images. As we can see, due to the high degree of intra-class variation of scene characters as well as the complexity of background, recognizing these text images is quite challenging even for state-of-the-art OCR methods.

In fact, characters are designed by humans and each category of characters has unique structure representing itself. Therefore, no matter how the background changes or the

Manuscript received June 14, 2013; revised September 5, 2013 and November 1, 2013; accepted December 18, 2013. Date of publication January 28, 2014; date of current version June 27, 2014. This work was supported by the Chinese National Natural Science Foundation under Projects 61172103, 61271429, 60933010, and 2012AA041312. This paper was recommended by Associate Editor P. L. Correia.

The authors are with the State Key Laboratory of Management and Control of Complex Systems, CASIA, Beijing 100190, China (e-mail: cunzhao.shi@ia.ac.cn; chunheng.wang@ia.ac.cn; baihua.xiao@ia.ac.cn; song.gao@ia.ac.cn; jinlong.hu@ia.ac.cn).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TCSVT.2014.2302522

character degrades, as long as the structure remains unchangeable, we could recognize them by detecting the unique structure from cluttered background. In other words, humans naturally make use of character-specific structure information when recognizing characters from scene images. Thus, a good scene character-recognition method should make use of both the local appearance and global structure information.

In this paper, we propose a novel scene text-recognition method combining structure-guided character detection and linguistic knowledge. As character detection and recognition is the premise for word recognition, it plays an important role for the overall performance. Thus, we propose a novel and effective character detection approach. Different from conventional multiscale sliding window character detection strategy, we use part-based tree-structure to model each category of characters so as to jointly detect and recognize characters. The characters could be recognized by detecting character-specific structures, seamlessly combining detection, and recognition together. As both the global structure and the local appearance informations contribute to the part-based tree-structured models (TSMs) for characters, the detection results are more reliable. To recognize the scene text, we combine the detection scores and language model into the posterior probability of character sequence from the Bayesian decision view. The final word-recognition result is obtained by maximizing the character sequence posterior probability via Viterbi algorithm. Since the purpose of word-recognition is to understand the words and the case insensitive results could satisfy this need, we only report case insensitive word-recognition results in this paper. We evaluate our method on a range of challenging data sets (ICDAR 2003, ICDAR 2011, SVT). Experimental results show that our method achieves state-of-the-art performance both for character detection and word recognition.

An earlier version of this paper has been presented in our conference paper [20]. This paper extends the previous version in several aspects: 1) more technical details of the TSMs for characters are given; 2) a new word-recognition framework is proposed so as to incorporate higher orders of linguistic knowledge; 3) more experiments and analysis on character detection and recognition based on TSM are given; 4) the robust reading performance without edit distance correction is evaluated and compared with state-of-the-art methods; and 5) and more analysis and discussions are given about the experimental results and the method. The rest of this paper is organized as follows. Section II briefly reviews the related work. Section III gives an overview of the proposed method. Section IV describes the character detection method, including the model, the inference, and the learning. Section V explains how to combine the detection scores and language model to recognize words from Bayesian decision view. Experimental results and discussions are given in Section VI and conclusion is drawn in Section VII.

## II. RELATED WORK

Most of the previous work on scene text recognition could be roughly classified into two categories: traditional OCR-based and object recognition-based methods.

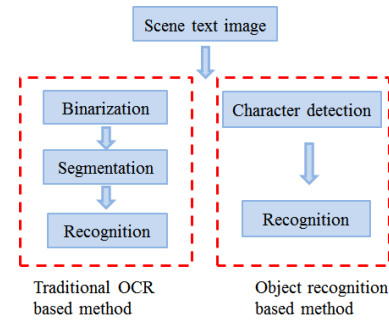


Fig. 2. Illustration of the traditional OCR-based method and object recognition-based method.

### A. Traditional OCR-based Methods

For traditional OCR-based methods as shown in Fig. 2, they focus on the binarization process, which segments text from background, and then the binary image could be segmented into individual characters, which could be recognized by the OCR engine. Various approaches have been proposed to binarize images with low quality or complex background. Gllavata *et al.* [21] first used a color quantizer to determine the color of text and background, and then adopted the modified k-means algorithm to classify pixels into text and background. Song *et al.* [22] used color-based k-means clustering to segment text from background. The performance of color clustering methods is dependent on the color consistency and also sensitive to noise and text resolution. Chen *et al.* [23] used mixtures of Gaussians to model the gray level distributions in the image and assigned the pixels to one of the Gaussian layer based on the prior of the contextual information modeled by a Markov random field. Ye *et al.* [24] proposed to train the Gaussian mixture models (GMM) of intensity and hue components in HSI color space using sampled pixels and utilized the GMM together with the spatial connectivity information to segment text pixels from the background. Li *et al.* [25] proposed to integrate local visual information and contextual label information into a conditional random field to segment text from complex background. Shi *et al.* [26] proposed to binarize video text images using graph cut algorithm based on the automatic acquired hard constraint seeds. Recently, Shivakumara *et al.* [27] introduced a novel ring radius transform and the concept of medial pixels on characters with broken contours in the edge domain for reconstruction to improve the character-recognition rate in video images. Feild *et al.* [28] proposed to use bilateral regression to model smooth color changes across an image region without being corrupted by neighboring image regions and use feedback from a recognition system to choose the best foreground region.

Compared with the conventional scanned document binarization methods such as Otsu [29] and Niblack [30], some recent binarization or preprocessing methods could indeed improve the scene text-recognition rates. However, since text in natural images has unconstrained resolution, illumination condition, size and font style, and the binarization results are unsatisfactory. Moreover, the loss of information during the



Fig. 3. Some scene text binarization results using [29]. The binarization results are quite disappointing, making it difficult for the following segmentation and recognition. (a) Scene text images. (b) Binarization result.

binarization process is almost irreversible, which means if the binarization result is poor; the chance of correctly recognizing the text is very small. As shown in Fig. 3, the binarization results are very disappointing, making it almost impossible for the following segmentation and recognition.

### B. Object Recognition-based Methods

On the other hand, object recognition-based methods assume that scene character recognition is quite similar to object recognition with a high degree of intra-class variation. For scene character recognition, these methods [13], [14], [31]–[34] directly extract features from original image and use various classifiers to recognize the character. Chen *et al.* [31] proposed a local intensity normalization method to handle lighting variations, then used a Gabor transform to obtain local features and finally adopted a linear discriminant analysis for feature selection. Weinman *et al.* [35] proposed to incorporate character appearance, bigram frequencies, similarity and lexicons into the recognition process. Smith *et al.* [36] also proposed to incorporate character similarity information to improve recognition performance. Based on bag-of-visual-words framework, De Campos *et al.* [32] benchmarked the performance of various features to assess the feasibility of posing the problem as an object-recognition task and showed that geometric Blur [37] and shape context [38] in conjunction with nearest neighbor (NN) classifier, performed better than other methods. Wang *et al.* [13] proposed to use histograms of oriented gradients (HOG) [39] in conjunction with an NN classifier and reported better performance. Newell and Griffin [33] proposed two extensions of HOG descriptor to include features at multiple scales and their method achieved promising performance on two data sets, chars74k [32] and ICDAR03-CH [2], using the same evaluation framework as De Campos [32]. Coates *et al.* [34] took an unsupervised approach to learn features from unlabeled data and the character-recognition results on the ICDAR03-CH [2] are quite promising.

While for object recognition-based scene text recognition, since there are no binarization and segmentation stages, as shown in Fig. 2, most of the existing methods [13]–[16] adopt multiscale sliding window strategy to get the candidate character detection results. Then, word-recognition strategies,

such as pictorial structures [13], [14] or CRF [15], [16] are used to get the final word-recognition results from the candidate character detection results. Elagouni *et al.* [40] adopted the convolutional neural networks (CNN) to get the candidate character detections and a graph model is used to determine the best sequence of characters. Novikova *et al.* [17] proposed to use maximally stable extremal regions as character candidates and formulate the problem of word recognition as the maximum *a posteriori* inference in a unified probabilistic framework. Recently, Wang *et al.* [41] used CNN to train the text detection and character-recognition modules and recognize the words with nonmaximal suppression (NMS) and beam search with the help of the lexicon. Weinman *et al.* [18] proposed to use probabilistic methods to coarsely binarize a given text region and jointly perform word and character segmentation during the recognition process, which achieved state-of-the-art performance.

### C. Structure-based Model for Object Detection

Structure-based model, which captures the local appearance properties and the deformable configuration of an object, has inspired great interest for object detection, since Felzenszwalb and Huttenlocher [42] proposed the pictorial structures framework for object recognition. In this framework, objects are represented by a collection of parts arranged in a deformable configuration, which has been proved to be effective for many applications. To deal with object by significant variations, Felzenszwalb *et al.* [43] proposed to use mixtures of star-structured model defined by a root filter plus a set of part filters and deformation models. Impressively, their method won the first place on PASCAL visual object detection challenge 2008 and 2009 [44], [45]. Later, Yang and Ramanan [46] proposed to detect articulated pose of human using flexible mixtures-of-parts. Tree structure is used to model co-occurrence and spatial relations, and the model could be efficiently optimized with dynamic programming. Recently, Zhu and Ramanan [47] proposed to jointly address the tasks of face detection, pose estimation, and landmark estimation using mixtures of trees with a shared pool of parts. Although their model is only trained with hundreds of faces, it compares favorably with the commercial systems trained with billions of examples.

Characters are designed by humans and each class of characters has a unique structure representing itself. Thus, we should try to use the global structure as well as local appearance information for character detection and recognition. The proposed approach is closely related to those methods in [46], [47]. We use modified part-based TSM to detect candidate characters.

## III. SYSTEM OVERVIEW

As mentioned above, a good scene character-recognition method should make use of both the local appearance and global structure information. Motivated by the recent progress in object detection [46], [47] using part-based TSM, we find that we could adopt these models to use both the global structure and local appearance information of characters.



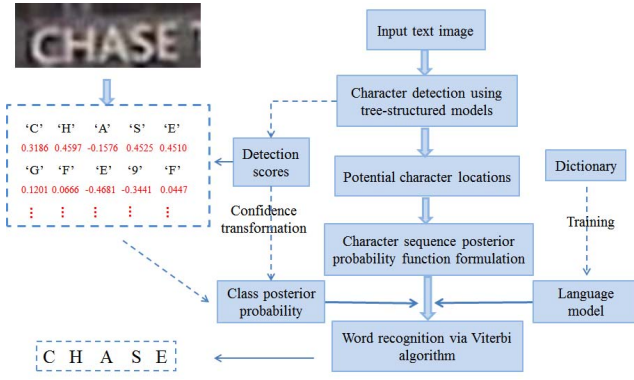


Fig. 4. Flowchart of the proposed system.

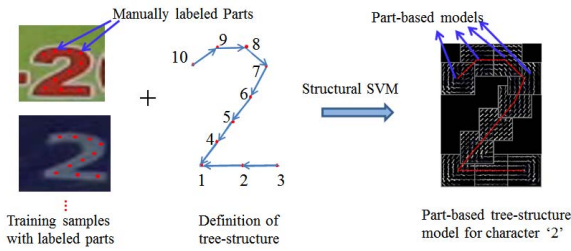


Fig. 5. Brief illustration of how to train the TSM for character 2. Red lines: topological relations of the parts and each rectangle corresponds to a part-based filter.

The flowchart of the proposed method is shown in Fig. 4. Given an input text image, first, we use part-based TSM for all the categories of characters to detect the character-specific structures, based on which we get the potential character locations. Then, we convert the candidate detection scores to posterior probabilities via confidence transformation. For word recognition, we combine the detection scores and language model into the posterior probability of the character sequence from the Bayesian decision view. Bigram, trigram, and even higher order language model could be incorporated. The final word-recognition result is obtained by finding the most probable character sequence via Viterbi algorithm.

#### IV. CHARACTER DETECTION USING PART-BASED TSM

We propose to recognize characters by detecting part-based tree-structures, which seamlessly combines detection and recognition together. Fig. 5 briefly shows how to train the TSM for character 2. Fig. 6(a) shows how to recognize the characters using the trained character models. Next, we will give details about the model, the inference of the character-specific structures and the learning of the parameters of the models.

##### A. Model

To make use of both global structure and local appearance information of characters, we build a part-based TSM for each category of characters.

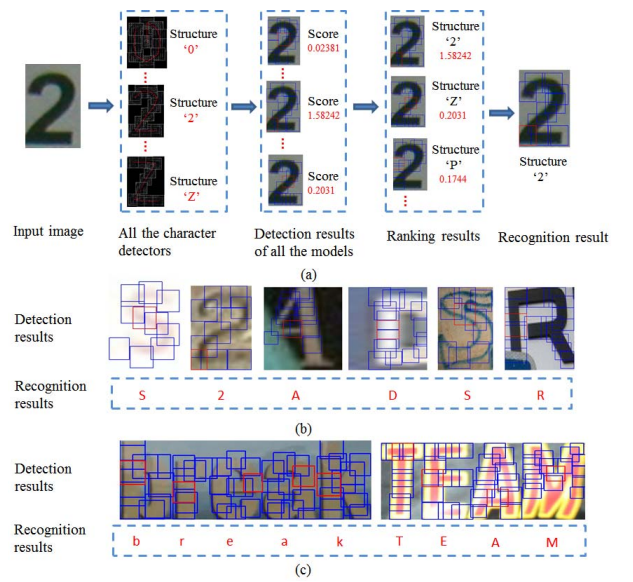


Fig. 6. Illustration of how to use the character detectors to recognize the characters and some detection and recognition results. The red rectangle labels the position of the root node of the tree while the blue ones label other parts of the character. (a) Illustration of how to recognize the character using the detectors. (b) Detection results of characters with contamination and deformation. (c) Detection results of text images after applying NMS.

1) *Model for Characters*: We represent each category of characters by a tree  $T_k = (V_k, E_k)$ , where  $k$  is the index of the model for different structures,  $V_k$  represents the nodes, and  $E_k$  specifies the topological relations of nodes [47]. Each node represents a part of the character. Let  $I$  represents the input image and  $l_i = (x_i, y_i)$  denotes the location of part  $i$ . Then, the score of the configuration of all the parts  $L = \{l_i, i \in V_k\}$  could be defined as

$$S(L, I, k) = S_{\text{App}}(L, I, k) + S_{\text{Str}}(L, k) + \alpha_k \quad (1)$$

where

$$S_{\text{App}}(L, I, k) = \sum_{i \in V_k} w_i^k \cdot \phi(I, l_i) \quad (2)$$

and

$$S_{\text{Str}}(L, k) = \sum_{ij \in E_k} w_{ij}^k \cdot \psi(l_i - l_j). \quad (3)$$

As we can observe, the total score of a configuration  $L$  for model  $k$  consists of the local appearance score in (2), the structure or shape score in (3), and the bias  $\alpha_k$ . Here,  $\alpha_k$  is a scalar bias or prior associated with character  $k$ . Next, we will give details about the appearance model and the shape model.

2) *Local Appearance Model*: Equation (2) is the local appearance model, which reflects the suitability of putting the part-based models on the corresponding positions.  $w_i^k$  represents the filter or the model for part  $i$ , structure  $k$ , and  $\phi(I, l_i)$  denotes the feature vector extracted from location  $l_i$ . Thus, the score of placing part  $w_i^k$  on position  $l_i$  is actually the filter response of template  $w_i^k$ . We choose HOG [39] as the local appearance descriptor due to its good performance on many computer vision tasks. For color image, we choose the

color channel with the largest gradient magnitude to calculate the HOG features.

3) *Global Structure Model*: Equation (3) is the structure or shape model, which scores the character-specific global structure arrangement of configuration  $L$ . Here, we set  $\psi(l_i - l_j) = [dx \ dx^2 \ dy \ dy^2]$ , where  $dx = x_i - x_j$  and  $dy = y_i - y_j$  are the relative distance from part  $i$  to part  $j$ . Each term in the sum acts as a spring that constrains the relative spatial positions between a pair of parts. The parameters  $w_{ij}^k$ , which are learned in the training process, could control the location of each part relative to its parent and the rigidity of each spring. By incorporating the elastic structure information, the model could detect characters with contamination or deformation, as shown in Fig. 6(b).

### B. Inferring Character-specific Structures

Inferring the character-specific structure corresponds to maximizing  $S(L, I, k)$  in (1) over all the possible configurations of all the parts,  $L$  and all the classes of characters  $k$ . Since the TSMs for all the characters are independent from each other, we could maximize  $S(L, I)$  for all the structures in parallel. Thus, for each structure, we need to maximize  $S(I)$  over  $L$

$$S^*(I) = \max_L S(I, L). \quad (4)$$

Since each structure  $T_k = (V_k, E_k)$  is a tree, the maximization for each structure could be done efficiently with dynamic programming. Let  $\text{child}(i)$  be the set of children of  $i$  in  $V_k$ . The messages that part  $i$  passes to its parent part  $j$  could be computed by the following:

$$m_{i \rightarrow j}(l_j) = \max_{l_i} (\text{score}_i(l_i) + w_{i,j} \cdot \psi(l_j - l_i)) \quad (5)$$

where

$$\text{score}_i(l_i) = w_i \phi(I, l_i) + \sum_{m \in \text{child}(i)} m_{m \rightarrow i}(l_i). \quad (6)$$

Equation (6) computes the local score of part  $i$ , which equals to the appearance score of part  $i$  plus the messages collected from the children of  $i$ . The message that part  $i$  passes to its parent part  $j$  is composed of: 1) the local score of part  $i$  and 2) the deformation penalty of parts  $i$  and  $j$ . Here, for each location of parent part  $j$ , we choose the highest score that part  $i$  could pass to part  $j$ . Fig. 7 shows an example of how we infer the parts as well as the tree-structure of character 2. We begin with the filter responses of the leaf nodes, 10 and 3, then the messages are passed to their parent parts 9 and 2, respectively, and when the messages are finally passed to the root node 1, we could get the scores of all the locations being the root. Then, we choose the root location with the highest score as the detected root part and backtrack to find the locations of other parts in the maximal configuration, as labeled by the red rectangles in Fig. 7.

1) *Rescoring*: Apart from unique structure, different parts of a character tend to have similar intensity, which we could use to further improve the performance. Thus, we rescore each

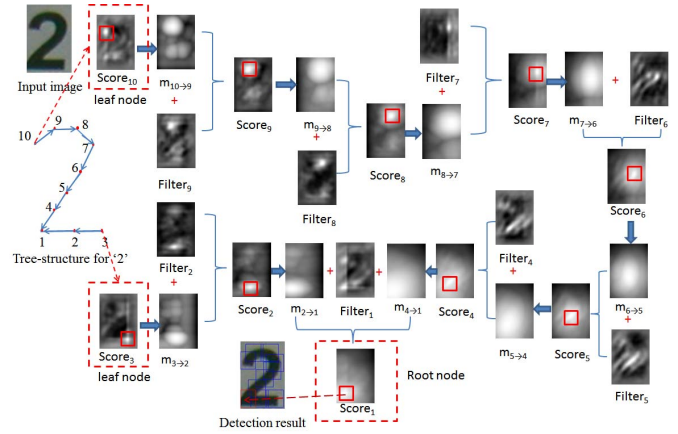


Fig. 7. Example of how to use the TSM to detect each part of the characters.  $\text{Filter}_i$  is the filter responses of part  $i$  on the input image.  $m_{i \rightarrow j}$  denotes the message being passed from node  $i$  to node  $j$ .  $\text{Score}_i$  represents the local score of part  $i$ . We begin with the filter responses of the leaf nodes, 10 and 3, then the messages are passed to their parent parts 9 and 2, respectively, and when the messages are finally passed to the root node 1, we could get the scores of all the locations being the root. Then, we choose the location with the highest score as the detected root part and backtrack to find the locations of all the parts in the maximal configuration as labeled by the red rectangles.

model by considering the intensity consistency of each part to the root part

$$S_{\text{new}}(L, I, k) = S(L, I, k) + \sum_{i=2}^N \delta_i \cdot \gamma D(i, 1) \quad (7)$$

where

$$D(i, 1) = \begin{cases} 0, & \text{if } \text{dis}_{i,1} < t \\ \text{dis}_{i,1}, & \text{if } \text{dis}_{i,1} \geq t. \end{cases} \quad (8)$$

$N$  refers to the number of parts,  $\text{dis}_{i,1}$  is the feature distance ( $l_1$  distance) between the part  $i$  and root part.  $\gamma$  is set to 0.2 by cross-validation.  $\delta_i = 1$  if part  $i$  is designed to be different from the root part and  $\delta_i = -1$  otherwise. Whether the part should be the same or different from the root part are revealed from the structure and part locations that we design for each class of characters. In the experiment, the threshold  $t$  is set to 1.1 by cross-validation. We choose histogram feature to reflect intensity consistency.

### C. Learning Model Parameters

For each category of characters, we construct a tree-structure-based model. To learn each model, we assume a fully supervised paradigm, where we are provided positive images with characters as well as part labels, and negative images without characters. Both the shape and appearance parameters are discriminatively learned using a structured prediction framework. First, we need to define the topological structure  $E_k$  for each model. Since digits and letters have simple and obvious structure, trees are natural for modeling them. We design the tree-structure for each category of characters by our experience and experimental results show that they perform quite well. Fig. 8 shows some examples of how we design structures for different classes of characters and how we label the parts. After designing the tree-structure for each

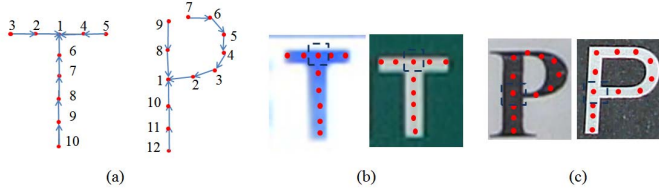


Fig. 8. Illustration of how to design tree structure for different characters and how to label the parts for training samples. (a) Tree structure for T and P, where red points are the nodes of the tree and 1 refers to the root node. (b) and (c) Examples of how to label the parts. We only label the centroid (red points) of each part and extract features from the regions shown in dashed rectangles to represent the parts.

category of characters, as shown in Fig. 8(a), we manually label the center of each part and extract features from the region centered by the node to represent the part, as shown in Fig. 8(b) and (c). There are two fundamental rules for us to design the structure and parts of characters. First, each node and its parent node should not be spatially too distant. For instance, in Fig. 8(a), for T, it is more suitable for node 5 to have node 4 as its parent than node 3, since node 4 is nearer. If we set node 3 as the parent of node 5, then the region we need to represent the node [the dashed rectangle in Fig. 8(b)] would be much larger and thus is not representable enough. Second, each node could only have one parent to prevent the loops. For instance, in Fig. 8(a), for T, if node 10 also has node 8 as its parent, the local loop 10, 9, 8 would appear. Similarly, if node 10 also has node 5 as parent, a larger loop 10, 9, 8, 7, 6, 5, 4, 1 would appear. While for P, if node 9 also has node 7 as its parent, the loop 1, 2, 3, 4, 5, 6, 7, 8, 9 would appear.

For a certain type of structure  $k$ , given labeled positive examples  $\{I_n, L_n, k_n\}$  and negative examples  $\{I_n\}$ , we define a structured prediction objective function similar to the one proposed in [46]. Let us write  $z_n = (L_n, k_n)$ . Note that, the scoring function in (1) is linear in model parameters  $(w, \alpha)$ . Concatenating these parameters into a single vector  $\beta$ , then we could write the score as

$$S(I, z) = \beta \cdot \Phi(I, z). \quad (9)$$

Now, we would learn a model of the following form:

$$\begin{aligned} \arg \min_{\beta, \xi_n \geq 0} \quad & \frac{1}{2} \beta \cdot \beta + C \sum_n \xi_n \\ \text{s.t.} \quad & \forall n \in \text{pos} \quad \beta \cdot \Phi(I_n, z_n) \geq 1 - \xi_n \\ & \forall n \in \text{neg}, \quad \forall z \quad \beta \cdot \Phi(I_n, z) \leq -1 + \xi_n. \end{aligned} \quad (10)$$

$C > 0$  is a constant that controls the tradeoff between the training error minimization and margin maximization. The above constraint states that positive examples should score better than 1 (the margin), while negative examples, for all configurations of part positions and structures, should score less than  $-1$ . The objective function penalizes violations of these constraints using slack variable  $\xi_n$ . The optimization of the above objective function is a quadratic program. The form of the learning problem is actually a structural support vector machine (SVM) and we use the dual coordinate descent solver [43] to solve the problem.

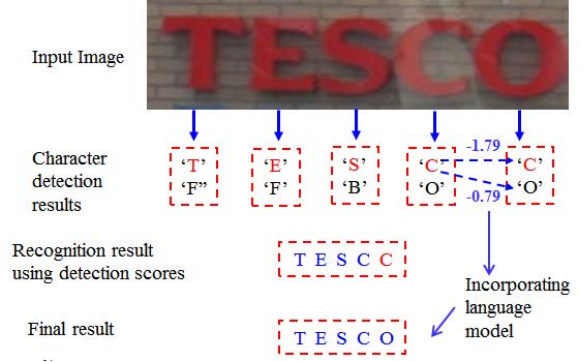


Fig. 9. Illustration of how to combine the detection scores and language model to recognize words. Given a text image, we first use TSMs to get the character detection results, based on which we get the potential character locations. If we only use detection scores to recognize the word, there might be ambiguities between similar characters. Thus, we use language model to distinguish the ambiguities.

Since the TSMs for different characters are independent from each other, we could learn the model parameters for each category of structures in parallel.

1) *Training Details:* Although both shape and appearance parameters are finally discriminatively learned using a structured prediction framework, before learning them jointly, we pretrain each part-based model to initialize the template parameters  $w_i$  in the TSM. While for the deformation parameters  $w_{i,j}$ , we initialize them to a constant. Since we focus on detecting and recognizing characters with certain structures, characters with similar structures such as, 0, O and o, P and p, K and k, X and x, should belong to the same class. Details of how we combine classes are listed in the rectangle below transfer with the same score in Fig. 10. Thus, in total, we have 49 classes of structures to detect and recognize. The number of positive training samples for each structure varies from 20 to 30, which are chosen from the Chars74k data set [32] and once the final structure-based model for each class is trained, they are used on all the tasks. We manually label the center of each part for all the images in the training set. All the training samples are normalized to the same height while preserving the aspect ratio. Both the part-based models and the overall TSM use the same negative training samples from ICDAR 2011 text localization training data set [48].

## V. WORD-RECOGNITION MODEL

The text images are normalized to the same height while preserving the aspect ratio. Since some lowercase character might only have half the size of the uppercase ones, image pyramid is used to deal with characters with different sizes. Given a scene text image, we only consider characters with the size from 0.5 to 1 of the normalized size. Although the character detection step provides us with a set of windows containing characters with high confidence, as shown in Fig. 6(c), inevitably it also produces some false positives and ambiguities between the similar characters. Fig. 9 shows an example. For the last character, both structures of C and O are detected. The score of C is even higher. If we only use the detection result to recognize the word, the results would



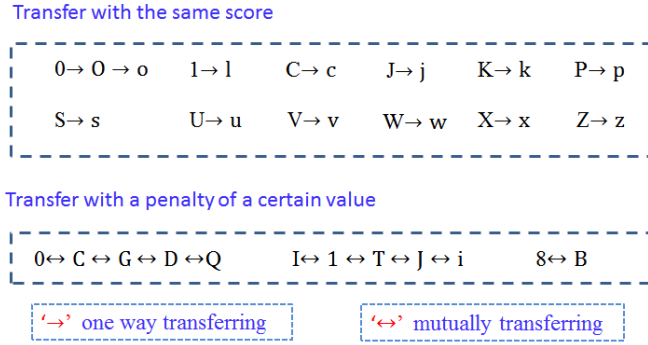


Fig. 10. Score transferring between classes. For classes with approximately the same structure, scores are directly transferred, while for classes with similar structures, scores are transferred with a penalty of a certain value.

be incorrect. Thus, we need to make use of other information, such as language model to eliminate these ambiguities. To this end, we combine the detection scores and language model from the Bayesian decision view. According to Bayesian decision under the 0/1 loss, given a text image  $X$ , MAP probability of character sequence  $C = \langle c_1, c_2, \dots, c_m \rangle$  is the optimal criterion for recognition [49], [50]. This posterior probability is formulated as

$$P(C|X) = P(C|X, l) \quad (11)$$

where  $l$  denotes the potential locations for characters. After applying NMS [47] on the original character detection results, the left detection windows constitute the potential locations. For NMS, we iterate over all windows in the image in descending order of their scores and if the first window has not yet been suppressed, we suppress all the other windows that have an overlap with it over some threshold. We set the overlap parameter for NMS to 0.3 in the experiment. Then, for location  $m$ , suppose we have detection  $i$  after NMS, we choose those detection windows, which are close to this location as the candidate characters for this location. The rules we used to choose the candidate characters are

$$O(R_i, R_j) > 0.5 \wedge D(e_i, e_j) < 0.23 \cdot h_i \quad (12)$$

where  $R_i$  and  $R_j$  are the bounding boxes of detection  $i$  and  $j$ , respectively.  $O(R_i, R_j)$  is the overlap ratio between the two bounding boxes.  $h_i$  represents the height of detection  $i$ .  $e_i$  and  $e_j$  are the horizontal axis of the center of detection  $i$  and  $j$ .  $D(e_i, e_j)$  is the  $l_1$  distance between the center of the two detections. We index the potential locations sequentially from left to right as  $\{1, 2, \dots, m\}$ . Equation (12) means if the overlap ratio of the bounding boxes of detection  $i$  and detection  $j$  is over a certain value and these two detections are horizontally near enough, we add detection  $j$  to the candidate characters for location  $m$ .

Given the potential character locations, the posterior probability of character sequence could be obtained using the Bayes' formula

$$P(C|X^l) = \frac{p(X^l|C)p(C)}{p(X^l)} \quad (13)$$

where the prior probability  $p(C) = p(c_1, \dots, c_m)$  is given by a statistical language model, which would be detailed in Section V-B. The likelihood function  $p(X^l|C)$  can be decomposed as

$$p(X^l|C) = \prod_{i=1}^m p(\mathbf{x}_i|c_i) \quad (14)$$

where we assume that the feature vectors  $x_i$  are only dependent on the corresponding character patterns. Similarly, we could get  $p(X^l)$  as

$$p(X^l) = \prod_{i=1}^m p(\mathbf{x}_i). \quad (15)$$

Consequently, substituting (14) and (15) into (13), we could get the posterior probability

$$P(C|X^l) = p(C) \prod_{i=1}^m \frac{p(\mathbf{x}_i|c_i)}{p(\mathbf{x}_i)} \quad (16)$$

which could be transformed to

$$P(C|X^l) = p(C) \prod_{i=1}^m \frac{p(c_i|\mathbf{x}_i)}{p(c_i)} \quad (17)$$

where the posterior probability  $p(c_i|\mathbf{x}_i)$  can be approximated using the character detection results (see Section V-A), and the prior probability  $p(c_i)$  is viewed as constants in classifier design, denoted by  $P$ . Substituting  $p(C) = \prod_{i=1}^m p(c_i|h_i)$  (see Section V-B) into (17) gives

$$P(C|X^l) = \prod_{i=1}^m \frac{p(c_i|\mathbf{x}_i)p(c_i|h_i)}{P}. \quad (18)$$

The two probabilistic terms in (18) correspond to the character-recognition model and the language model, respectively. However, (18) does not consider the different contribution of the character model and language model. In the following, we take the logarithm of probability and introduce a tradeoff parameter  $\lambda$  to get a generalized objective function as

$$\begin{aligned} f(C, X^l) &= \log(P(C|X^l)) \\ &= \sum_{i=1}^m (\log(p(c_i|\mathbf{x}_i)) + \lambda \log(p(c_i|h_i))) + m \log \frac{1}{P}. \end{aligned} \quad (19)$$

#### A. Character-recognition Model

The character-recognition posterior probability  $p(c|\mathbf{x})$ , forms the basis for word recognition. As we only have 49 categories of TSMs for characters and there are 62 classes to recognize, we transfer detection results of some classes to those ones that have similar structures. Fig. 10 shows how we transfer the scores. For characters with approximately the same structures, such as uppercase and lowercase, X and x, P and p, the scores of one class is directly transferred to other classes with the same score. As a matter of fact, during training, we combine samples from class 0, O, and o to form a new class 0. As we can see the list of transfer with the

same score in Fig. 10, there are 13 classes being combined to other classes, thus resulting 49 classes in total. For characters with similar structures, such as Q, O, and G, the scores of one class are transferred to another class with a penalty of a certain value (set to 0.2 in the experiment). The TSMs for characters could only provide us with character detection scores, which are not class posterior probabilities. Thus, we resort to confidence transformation methods to convert the detection scores to posterior probabilities [49]. The confidence transformation could make the detection scores of different character detectors comparable with some extent.

Here, we choose the soft-max function for probabilistic confidence transformation. The soft-max function is defined as

$$P(c_j|\mathbf{x}) = \frac{\exp(-\theta d_j(\mathbf{x}))}{\sum_{i=1}^M \exp(-\theta d_i(\mathbf{x}))} \quad (20)$$

where  $M$  is the total number of defined classes,  $d_j(\mathbf{x})$  is the dissimilarity score of class  $c_j$  output by the character detector, and  $\theta$  is the confidence parameters. For the dissimilarity score  $d_j$ , we set it to the negative of the detection score in (7). However, the soft-max function forces the sum of posterior probabilities to one even for noncharacter (outlier) patterns. Similar to [49], we also introduce an outlier class dissimilarity score  $d_0(\mathbf{x}) = \eta/\theta$  in soft-max confidence, and the probabilities can be formulated as

$$P(c_j|\mathbf{x}) = \frac{\exp(-\theta d_j(\mathbf{x}) + \eta)}{1 + \sum_{i=1}^M \exp(-\theta d_i(\mathbf{x}) + \eta)}. \quad (21)$$

After getting multiclass probabilities, the probability of outlier class is

$$P(c_{\text{outlier}}|\mathbf{x}) = \frac{1}{1 + \sum_{i=1}^M \exp(-\theta d_i(\mathbf{x}) + \eta)}. \quad (22)$$

The confidence parameters are optimized by minimizing the cross entropy loss function on a validation data set (preferably different from the data set for training the TSMs).

### B. Statistic Language Model

In character sequence inference or word recognition, the statistical language model is used to give the prior probability of a certain character sequence. If the sequence  $C$  contains  $m$  characters,  $p(C)$  can be decomposed as

$$p(C) = \prod_{i=1}^m p(c_i|c_1^{i-1}) = \prod_{i=1}^m p(c_i|h_i) \quad (23)$$

where  $h_i = c_1^{i-1} = \langle c_1 \dots c_{i-1} \rangle$  denotes the history of character  $c_i$ . An  $n$ -gram model only considers the  $n-1$  history characters

$$p(C) = \prod_{i=1}^m p(c_i|c_{i-n+1}^{i-1}) \quad (24)$$

where  $n$  is called the order of the model. The character bigram and trigram are usually used

$$p_{\text{bi}}(C) = \prod_{i=1}^m p(c_i|c_{i-1}) \quad (25)$$

$$p_{\text{tri}}(C) = \prod_{i=1}^m p(c_i|c_{i-2}c_{i-1}). \quad (26)$$

We use the Stanford Research Institute Language Modeling Toolkit [51] to learn the parameters of  $n$ -gram models in a large English dictionary with around 0.5 million words provided by Weinman *et al.* [35]. The default smoothing technique (Katz smoothing) and the entropy-based pruning are used.

Since some potential locations might not contain any characters, we use  $\epsilon$  to represent a null label. We define the bigram as

$$p(c_j|c_i) = \begin{cases} 0.35, & \text{if } c_i \leq 10 \wedge c_j \leq 10 \\ \text{trans}_2, & \text{if } c_i \neq \epsilon \wedge c_j \neq \epsilon \\ 0.03, & \text{otherwise} \end{cases} \quad (27)$$

where  $\text{trans}_2$  is the bigram joint occurrence probability of characters ( $c_j, c_i$ ) learnt from the dictionary. The above definition increases the joint occurrence probability of two digits so that the algorithm could deal with the situation in which all the characters are digits. The above definition also keeps one of the characters from being assigned a null label  $\epsilon$  unless the posterior probability of the outlier class is very high. Besides, we also give a low joint occurrence probability of digits and characters to prevent digits from occurring in the word. The trigram is defined as

$$p(c_k|c_i c_j) = \begin{cases} 0.35, & \text{if } c_i \leq 10 \wedge c_j \leq 10 \wedge c_k \leq 10 \\ \text{trans}_3, & \text{if } c_i \neq \epsilon \wedge c_j \neq \epsilon \wedge c_k \neq \epsilon \\ \mu \cdot p(c_k|c_j), & \text{if } c_i = \epsilon \\ \mu \cdot p(c_k|c_i), & \text{if } c_j = \epsilon \\ \mu \cdot p(c_j|c_i), & \text{if } c_k = \epsilon \end{cases} \quad (28)$$

where  $\mu$  is set to 0.35 by cross-validation, and  $\text{trans}_3$  is the trigram joint occurrence probability of characters ( $c_k, c_i, c_j$ ) learnt from the dictionary. The joint occurrence probability of three digits is also assigned a higher value so that it could deal with words which only contain digits. The above definition means that if one of the characters has a null label  $\epsilon$ , then the probability of the joint occurrences of three characters is in proportion to the joint occurrence of the remaining two characters. The higher order linguistic terms are defined in similar fashion.

### C. Inferring the Word

Inferring the word corresponds to finding the labeling sequence that maximizes (19) as

$$C^* = \underset{C}{\text{argmax}} \sum_{i=1}^m (\log(p(c_i|\mathbf{x}_i)) + \lambda \log(p(c_i|h_i))) + m \log \frac{1}{P}. \quad (29)$$

Note that,  $m \log \frac{1}{P}$  is a constant. Thus, we only need to consider the character and language models. We use Viterbi algorithm to find the best character sequence.



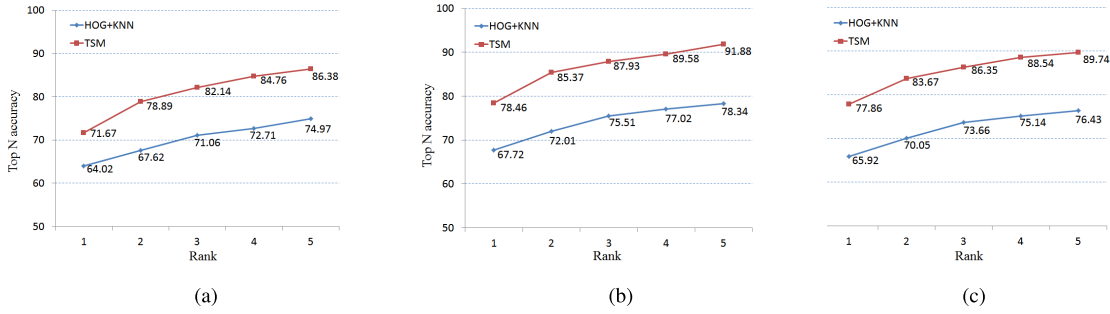


Fig. 11. Character-recognition results with different candidate number on different data sets. Since we focus on detecting characters with unique structures, we only train 49 classes of character model whose structures are different from each other. Thus, the recognition results are reported on 49 classes. Top N accuracy gives the cumulative accuracies of top N ranks. (a) Chars74k dataset(%). (b) ICDAR-03-CH-train dataset(%). (c) ICDAR03-CH-test dataset(%).

## VI. EXPERIMENTAL RESULTS AND DISCUSSION

In this section, we give detailed evaluation of the proposed character detection and word-recognition method. Since character detection and recognition are the basis for word recognition, we first evaluate the detection-based character-recognition method. We also compare the proposed character detection method with state-of-the-art sliding window-based detection methods. In addition, we compare the proposed TSM with deformable part-based model (DPM), which treats part locations as latent variables. For word-recognition task, we compare our results with state-of-the-art methods [13]–[20], [28], [41] for both robust reading and word spotting tasks.

### A. Data Sets

1) *Character-recognition Data Sets*: To evaluate the performance of the proposed detection-based character-recognition method, we test the recognition rate on two public data sets: Chars74k [32] and ICDAR 2003 robust character-recognition data set (ICDAR03-CH) [2]. These data sets contain cropped character images.

Since we choose training samples for all the structures from Chars74k data set, all the remaining images except those chosen as training samples comprise the test set for chars74k data set. While for ICDAR03-CH data set, since we do not use the training set to learn the model parameters, we evaluate the performance on both the training and test sets. In total, we have 6148, 5835, and 5245 test samples for Chars74k, ICDAR03-CH-Train and ICDAR03-CH-Test, respectively.

2) *Word-recognition Data Sets*: We use the challenging public data sets street view text (SVT) [13], ICDAR 2003 robust word recognition (ICDAR03) [2] and ICDAR 2011 word recognition (ICDAR11) data sets [48] to evaluate the performance of the overall word-recognition method. The SVT data set contains images taken from Google street view. Since we focus on the word-recognition task, we use the SVT-WORD data set following the experimental protocol of [13] and [14].

### B. Detection-based Character Recognition

To recognize characters using the detection model, we apply each character-specific TSM on the image and choose the structure with the highest score as the recognition result.

1) *Recognition Results With Different Candidate Numbers*: Apart from the correct recognition accuracy (top 1 accuracy), we also report the top N accuracy, which gives the cumulative accuracies of top N ranks (N varies from 2 to 5). As we only have 49 classes of character models whereas most of the previous methods [13], [14], [33] have 62 classes, we could not directly compare our method with these methods. However, similar to most of the methods, we also choose HOG+KNN as the benchmark method. Thus, the readers could compare our method with previous methods in a certain way. For HOG+KNN, each image is partitioned into  $4 \times 3$  blocks, from which we extract HOG [39] features, and KNN is used to recognize the character. The recognition results on Chars74k, ICDAR03-CH-Train, and ICDAR03-CH-Test are shown in Fig. 11.

The results show that the proposed TSM outperforms HOG+KNN by more than 10% on ICDAR03-CH data set, which reduces the error rate by more than 30%, when only considering the first candidate. When we consider the second best result, we find that the performance of TSM improves more quickly with an increase of about 8% whereas the recognition rate of HOG+KNN only increases 3%–5%. The results suggest: 1) the effectiveness of the TSMs, as they tend to detect and recognize characters with certain structures and 2) the high possibility of achieving better word-recognition result if we incorporate linguistic knowledge to deal with ambiguities between similar structures. When we consider the recognition rates of the first five candidates, the result is quite encouraging, reaching 86.38%, 91.88%, and 89.74% on Chars74k, ICDAR03-CH-Train, and ICDAR03-CH-Test, respectively. The recognition rate on Chars74k is lower, mainly because the remaining samples are generally harder to recognize after we choose the better ones as the training samples. Besides, since all the training samples are chosen from Chars74k data set, the results further demonstrate that, for the proposed method, the model trained on one data set could generalize well on other data sets, achieving recognition rates of 91.88% and 89.74% on ICDAR03-CH-Train and ICDAR03-CH-Test, respectively.

2) *Recognition Results With Different Training Samples*: We also evaluate the performance of TSM using different training samples, following the evaluation framework described in [32] and [52]. For Chars74k, we choose five and 15 training

TABLE I  
RECOGNITION RESULTS WITH DIFFERENT TRAINING SAMPLES  
FOR 49 CLASSES ON CHARS74k(%)

Method	Chars74k-5	Chars74k-15
Yi and Tian [52]	53.2	68.3
Our method	62.76	72.70

samples per class, referred to Chars74k-5 and charas74k-15, respectively. The test set contains 15 images per class. The training and test samples are randomly chosen and we report the average recognition results over 10 runs. Since we only have 49 classes of character models, the results are for 49 classes. We compare our results with state-of-the-art method proposed in [52]. From [52], we obtained the results of 49 classes after combining some classes into one class according to the list of transfer with the same score in Fig. 10.

The 49-class recognition results are shown in Table I. The results show that our method outperforms the method proposed in [52] considerably both for Chars74k-5 and Chars74k-15. When 15 images per class are used for training, TSM outperforms their method by about 5%, which reduces the error rate by about 14%. However, when we use five training samples per class, our method outperforms their method by more than 9%, which reduces the error rate by 19.6%, demonstrating the effectiveness of TSM in the face of limited training samples. As our TSM imposes strict independence assumptions between different parts of characters, it is more beneficial when training data are limited.

### C. Character Detection

To evaluate the superiority of the proposed character detection method over conventional sliding window-based detection strategy for word recognition, we compare the character detection results using TSM+NMS with state-of-the-art character classifier CNN [41] using sliding window-based detection and NMS. For CNN, we downloaded the code from Wang's homepage<sup>1</sup> and used the CNN model trained by Wang *et al.* We implemented the character detection based on the provided code and informed the authors of the experiment. We choose ICDAR 2003 cropped word-recognition data set as the benchmark data set. All the images in the test set, including those with less than two characters or with nonalphanumeric characters, are used for character detection. Similar to the evaluation methods of PASCAL VOC [44], [45], we compute the intersection over union measure of a detected window compared with the ground truth to evaluate the detection performance. A detection box is counted as a match if it overlaps a ground truth bounding box by more than 50% and the characters also match. We use the precision and recall (P/R) curves acquired by sweeping over a threshold on the detection scores to show the results.

Both the case-sensitive and case-insensitive results are evaluated and P/R curves are shown in Fig. 12. The best f-scores and the corresponding recall and precision are listed in the brackets next to the name of the detection methods

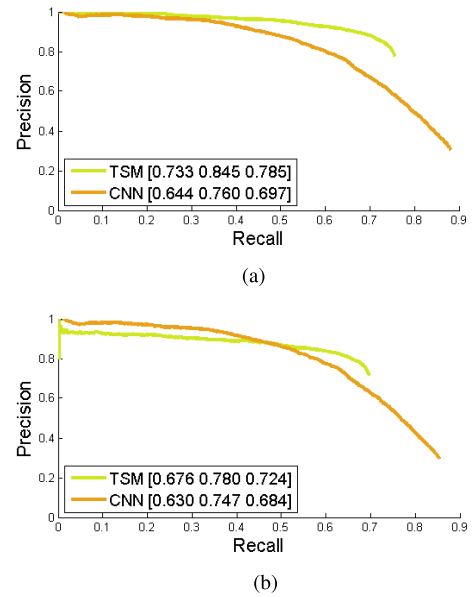


Fig. 12. P/R curves of TSM and CNN for character detection on ICDAR 2003 word-recognition data set. The recall, precision and f-score at the best f-score are shown in brackets next to the name of the method. (a) Case insensitive. (b) Case sensitive.

in the order of recall, precision and f. We also test the P/R curves of TSM and CNN without NMS; the best f-scores of both methods are not as good as those with NMS. Thus, we only report the results with NMS here. Since we apply NMS on the original detection results and we only keep the detection windows with the highest score at each potential character location, the P/R curves stop before the precision goes to 0. The case-insensitive results are shown in Fig. 12(a). The results show that the proposed TSM outperforms CNN considerably both for precision and recall. For the case-sensitive results shown in Fig. 12(b), TSM also outperforms CNN at the best f-scores. The results demonstrate that TSM, which integrates character detection and recognition together is more effective for character detection. However, the superiority over sliding window-based detection method using CNN of case-sensitive is not as much as that of case-insensitive. This is reasonable, since we only have 49 classes of character detectors. Some uppercase and lowercase characters could not be distinguished by our character detectors.

We also report the word-recognition result using the word spotting strategy PLEX from [14] to evaluate the performance of the proposed character detection method. Pictorial structure is used to represent each word in the lexicon. In this case, based on the character detection results of the proposed TSM and the SYNTH+FERNS proposed in [14], same word spotting strategy PLEX is used to find the final word. Similar to [14], we ignore words with less than two characters or with nonalphanumeric characters. For ICDAR03, we measure the performance using a lexicon created from all the words that appear in the test set [we call this ICDAR03(FULL)], and with lexicons consisting of the ground truth words for that image plus 50 random words added from the test set [we call this ICDAR03(50)]. In the SVT-WD case, a lexicon of about

<sup>1</sup><http://ai.stanford.edu/~twangcat/>

TABLE II  
WORD-RECOGNITION RESULTS USING WORD SPOTTING  
STRATEGY PLEX

Method	FERNs+PLEX [14]	TSM+PLEX
ICDAR03(FULL)	62	70.47
ICDAR03(50)	76	80.70
SVT	57	69.51

50 words is provided with each image as part of the data set. The word-recognition results are shown in Table II.

The results demonstrate that TSM+PLEX outperforms FERNs+PLEX considerably on all the tasks. Since we use the same word spotting strategy PLEX, the only difference between the two methods lies in the character detection method. Wang *et al.* [14] used Ferns to detect and recognize characters in the sliding window fashion, which only implicitly uses global character-specific structure information and is not elastic enough as the TSM. While for the proposed character detection method, we recognize characters by detecting character-specific structures, which seamlessly combine detection and recognition together. Furthermore, since our detection model could make use of both global structure information and local appearance information, the detection results are more reliable and representative.

#### D. TSM Versus DPM

We compare the proposed TSMs, which need class labels as well as part labels with DPM, which treat part locations as latent variables. For DPM, we first train the character models using the same training samples as TSM, labeled as DPM-1. We also train the DPMs using the training samples for TSM plus some negative training samples, labeled as DPM-2. Concretely, for class  $k$ , we add all the positive training samples from other classes as negative samples for class  $k$ . To evaluate the performance of DPM with more positive training samples, we use all the images from Chars74k data set as training samples, labeled as DPM-All. The 49 class-based recognition results of TSM, DPM-1, DPM-2, and DPM-All on ICDAR03-CH-Test are shown in Table III. The results show that TSM outperforms DPM-1, DPM-2, and DPM-all significantly. Since DPM-2 has more negative training samples to deal with the ambiguities between different classes, it outperforms DPM-1 several percents. However, its recognition accuracy is still much lower than that of TSM. When we use all the images from Chars74k to train DPM, the performance could be further improved. However, even with the extra positive training samples, DPM still does not perform as well as the TSM. We argue that the main reasons lie in: 1) the tree-structures for characters, which are designed by human are more representative than the structure learnt by DPM; 2) we could not use the rescoring strategy to further improve the performance since the part locations are latent variables for DPM; and 3) the fully supervised learning of TSM are more effective than DPM, which treat part locations as latent variables especially when the training samples are limited.

We also evaluate the character detection performance of DPM-1 and DPM-2 in the term of P-R curves. We use the

TABLE III  
49-CLASS RECOGNITION RESULTS OF TSM, DPM-1, DPM-2, AND  
DPM-ALL ON ICDAR03-CH-TEST DATA SET(%)

Models	TSM	DPM-1	DPM-2	DPM-All
Accuracy	77.86	58.89	66.11	69.23

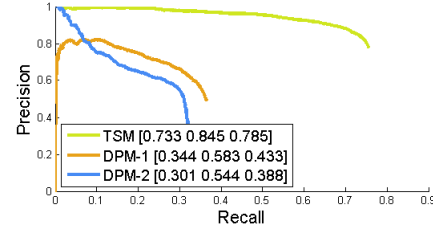


Fig. 13. P/R curves of TSM, DPM-1, and DPM-2 for character detection on ICDAR 2003 word-recognition data set. The recall, precision and f-score at the best f-score are shown in brackets next to the name of the method.

TABLE IV  
CASE-INSENSITIVE RECOGNITION RESULTS WITH DIFFERENT ORDERS  
OF LANGUAGE MODEL ON ICDAR 2003 DATA SET

The order of language model	2	3	4	5
Accuracy	50.46	58.56	56.48	50.28

whole ICDAR 2003 cropped word-recognition data set as the benchmark data set. The P-R curves of TSM, DPM-1, and DPM-2 are shown in Fig. 13. The results demonstrate that for character detection, TSM has greater superiority over DPM both for recall and precision. The best f-score for TSM is 78.5%, whereas for DPM-1 and DPM-2, the f-scores are only 43.3% and 38.8%.

#### E. Word Recognition

To recognize the word, we combine the detection scores and language model from the Bayesian decision view. We use ICDAR03-CH-Train to learn the confidence transformation parameters for the class posterior probabilities. We use ICDAR03, ICDAR11 and SVT data sets to evaluate the proposed word-recognition method. Same n-gram language model learnt from the lexicon with 0.5 million words is used on all the data sets.

1) *Evaluation Methods*: We evaluate the word recognition result with and without edit distance correction: 1) on one hand, we report recognition accuracies without edit distance correction using n-gram models. In this case, the word recognition results are the raw character sequence inferred using the Viterbi algorithm and 2) on the other hand, similar to the evaluation scheme in [14] and [16], we use the inferred result to retrieve the word with the smallest edit distance in the lexicon. Here, we use the normalized edit distance (calculated as the edit distance between the inferred word and the lexicon word divided by the length of the lexicon word) to deal with the possible bias toward shorter words. For ICDAR data sets, we measure performance using a lexicon created from all the words in the test set [ICDAR03(FULL), ICDAR11(FULL)], and with lexicon consisting of the ground truth words plus 50 random



Fig. 14. Evaluation of different contribution of the character model and language model. Trigram is used and we report the case-insensitive results without edit distance correction on ICDAR03 data set.


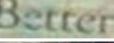




Text image	Without language model	Bi-gram	Tri-gram
	JUN9IE	JUNGIE	JUNGLE
	SB1P	SBIP	SHIP
	BETTCR	BETROR	BETTOR
	QUEEN	OLEEN	QUEEN
	FOPUM	FORUM	FORUM
	GGLGHESTER	GGICHESTER	COLCHESTER
	F1SH8MAN	FISHYMAN	FISHMAN

Fig. 15. Comparison results of the proposed method without language model, with bigram and trigram. Characters in red represent incorrect recognition. Without language model, one or two characters might be incorrectly recognized. With bigram, some ambiguities between similar characters could be distinguished. With trigram, most of the results are satisfactory.

words from the test set [ICDAR03(50), ICDAR11(50)]. For SVT data set, we use the lexicon provided in [14].

2) *Order of the Language Model*: We report the case-insensitive recognition accuracy varying the order of language model from 2 to 5. The results on ICDAR 2003 data set are shown in Table IV. The results show that when the order is 3, the system acquires the best performance. When the order is 5, the performance drops considerably. The results are quite reasonable, since increasing the order forces the recognized word to be a word from the dictionary, which would cause poor performance for nondictionary word. What's more, as we only use 0.5 million words to train the language model, the 4 or higher order language model might not be well trained. For instance, given the nondictionary word WIVENHOE from the test set, the recognition result with trigram is correct, whereas the result with four-gram is WIVONNOG. Since the transition probability of  $P(O|WIV)$  learnt from the dictionary is much higher than that of  $P(E|WIV)$ , the result is incorrect from the fourth letter. In the following section, the order is set to 3.

3) *Different Contribution of the Character and Language Models*: Since the final posterior probability of character sequence is composed of the character and language models, first we need to decide parameter  $\lambda$ , which controls the different contribution of the two models. Varying  $\lambda$  from 0 to 0.9 with a step of 0.1, we report the case-insensitive

TABLE V  
WORD-RECOGNITION RATES OF THE PROPOSED METHOD AND RECENT STATE-OF-THE-ART METHODS ON ICDAR 2003, ICDAR 2011, AND SVT IN ROBUST READING SCENARIO (WITHOUT EDIT DISTANCE CORRECTION)

Method	ICDAR03	ICDAR11	SVT
<b>case insensitive</b>			
Mishra <i>et al.</i> [15] (bi-gram)	45	-	23.49
Mishra <i>et al.</i> [15] (4-gram)	57.92	-	49.46
Novikova <i>et al.</i> [17]	-	66.7	-
Our method (bi-gram) (Case-2)	50.64	50.96	29.37
Our method (tri-gram) (Case-1)	52.73	-	-
Our method (tri-gram) (Case-2)	58.56	55.91	38.64
<b>case sensitive</b>			
Feild <i>et al.</i> [19]	52.90	41.04	-
Feild <i>et al.</i> [19] (with error correction)	62.76	48.86	-
Weinman <i>et al.</i> [18]	-	57.7	-

word-recognition result using trigram language model without edit distance correction on ICDAR03 data set. The results are shown in Fig. 14. From the results we can observe that, when we only use the character model to recognize the word ( $\lambda$  is set to 0), the recognition rate is only 29%. This demonstrates the importance of using the language model to distinguish similar characters. The results also show that when  $\lambda$  is 0.2 the proposed method gives the best performance. In the following experiments, we set  $\lambda$  to 0.2. Fig. 15 shows some comparing results of the proposed method without language model, with bigram and trigram language model. Without language model, one or two character might be incorrectly recognized. Bigram and trigram could help to distinguish ambiguities between the similar characters.

4) *Robust Reading Scenario (Without Edit Distance Correction)*: We report the open vocabulary word recognition results without edit distance correction. The recognition accuracies of our method along with those of the state-of-the-art methods are shown in Table V. For ICDAR 2003 test set, we give the results on Cases 1 and -2 for future comparison. For Case-1, all the 1065 images in the test set constitute the test set. While for Case-2, we ignore words with less than two characters or with nonalphanumeric characters. We listed the case-sensitive and-insensitive results separately in Table V. For our method, all the results are case insensitive, since we only have 49 classes of character models and thus some uppercase and lowercase characters are indistinguishable.

From the results we can observe that, when bigram is used, the proposed method outperforms Mishra *et al.* [15] by 5% on ICDAR03 and SVT. Furthermore, the proposed method with trigram language model also outperforms Mishra *et al.* [15], which used four-gram language model on ICDAR03 data set. Our method also outperforms the method proposed in [19] by 6% without web-based error correction. However, with web-based error correction, their method improves significantly by about 10%. Since our recognition accuracy of 58.56% is also the raw inference result from the Viterbi algorithm, the web-based error correction could be easily applied on our method to further improve the performance. We also report the recognition accuracy on ICDAR11 data set for future comparison. The method proposed by Novikova *et al.*





Fig. 16. Examples from SVT that our method failed to recognize.

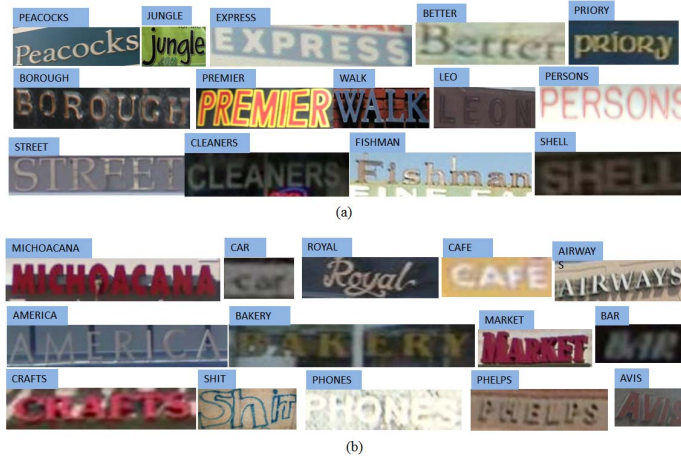


Fig. 17. Examples of word-recognition results of the proposed method. (a) Examples of scene text images being correctly recognized without edit distance correction. (b) Examples of scene text images being correctly recognized with edit distance correction. The results show that our method could recognize scene text with low resolution, different fonts, and distortions.

outperforms our method considerably on ICDAR 2011 data set. However, they added all the words in the test set to the lexicon, which might help to improve the results. With lexicon-based parsing, method in [18] outperforms our method on ICDAR 2011. Some of our recognition results are shown in Fig. 17(a). Our method does not perform well enough on SVT data set. Although our character detectors could detect characters with certain deformation or distortion, as shown in Fig. 6, it would fail to detect characters with large deformation or contamination, which is quite common on SVT data set, as shown in Fig. 16.

5) *Word Spotting Scenario (With Edit Distance Correction)*: We also compare our method with state-of-the-art methods [14]–[16], [40] in the word spotting scenario following the experimental protocol of [13], [14]. We use the edit distance correction strategy to retrieve the words with the smallest edit distance. However, sometimes there exist several words in the lexicon, which have the same edit distance with the inferred results. In this case, we use the detection scores of each character of the words to choose the one with the highest total scores as the recognition result. Table VI shows the results of our method and the state-of-the-art ones.

The proposed method outperforms recent proposed binarization-based method [28] by more than 6% on ICDAR03 (about 26% in error reduction), by more than 15% on ICDAR11 (about 55% in error reduction), and by more than 20% on SVT (about 41% in error reduction). Although the binarization method proposed in [28] did improve the segmentation results for some scene text images, some binarization

results are still unsatisfactory due to unconstrained lighting conditions, variation in font and color, low resolution, and complex background of scene text images.

Our approach also outperforms TSM+PLEX by 6%–9%, showing the effectiveness of the word-recognition model, which incorporates detection scores and linguistic knowledge, since both methods adopt the same character detection method TSM. Our method also outperforms the approach proposed in [16] and [15] by about 10% on ICDAR03(FULL) and ICDAR03(50), which is about 30% in error reduction. Note that Mishra *et al.* used the CRF model to encode character detection results and language model. However, they used the multiscale sliding window strategy to get the candidate character locations and SVM to classify these characters. The detection method is not as good as the proposed tree-structured character detection method, which combines detection and recognition together. Furthermore, they built the CRF model on all the detection windows as long as their spatial distance and overlap ratio satisfy a certain condition, which makes the CRF model more complex than our word-recognition model since our posterior probability is defined on the potential character locations. What's more, Mishra *et al.* [16] computed the node-specific lexicon prior for each text image from their corresponding lexicon, which means: 1) the lexicon priors heavily rely on the lexicon for that image and 2) the computation cost for each image is increased since the lexicon prior should be recomputed for each image. On the contrary, we use the same n-gram language model learnt from a large dictionary with 0.5 million words on all the tasks. This further demonstrates the generalization ability and the adaptivity of the proposed method.

The proposed method compares favorably with the method proposed in [41], which leverages large, multilayer CNNs to train the detection and recognition modules. However, a large amount of training samples are needed to train the CNNs and the test samples should not differ a lot from the training samples. Since Wang's training data set is chosen from ICDAR03 and Chars74k data sets, the results on SVT are relatively lower. While for the proposed TSM, only 20–30 training samples are used to train each model and the model trained on one data set performs satisfactorily on other ones. The method proposed by Weinman *et al.* outperforms our method by 4% on SVT data set, which is quite promising. However, they synthesized  $1866 \times 4 = 7464$  images for each class of characters, whereas we only have 30 training samples per class.

Since most of the text detection methods could not give perfect detection bounding boxes, meaning that the detected region also contains extra background pixels. To show the effectiveness of the proposed method to recognize scene text that is detected by various text detection methods, we extended the cropped word image in the original image to left, right, up, and bottom to get a bigger text image that is similar to text regions acquired from text detection methods. The results on these extended images are almost the same with those on the cropped word image, showing that our text-recognition method is quite suitable for recognizing detected text images. Thus, our method could be easily combined with existing text detection methods to robustly read text in natural images.

TABLE VI  
WORD-RECOGNITION RATES OF THE PROPOSED METHOD AND RECENT STATE-OF-THE-ART METHODS ON  
ICDAR 2003, ICDAR 2011, AND SVT IN WORD SPOTTING SCENARIO (WITH EDIT DISTANCE CORRECTION)

Method	ICDAR03(FULL)	ICDAR03(50)	ICDAR11(FULL)	ICDAR11(50)	SVT
ABBY9.0 [53]	55	56	-	-	35
SYNTH+PLEX [14]	62	76	-	-	57
TSM+PLEX (this paper)	70.47	80.70	74.23	80.25	69.51
Mishra <i>et al.</i> [16]	-	81.78	-	-	73.26
Mishra <i>et al.</i> [15]	67.79	81.78	-	-	73.26
Elagouni <i>et al.</i> [40]	66.19	-	-	-	-
Feild and Learned-Miller [28]	73.43	79.47	62.28	72.69	54.20
Shi <i>et al.</i> [20]	79.30	87.44	82.87	87.04	73.51
Novikova <i>et al.</i> [17]	82.8	-	-	-	72.9
Wang <i>et al.</i> [41]	<b>84</b>	<b>90</b>	-	-	70
Weinman <i>et al.</i> [18]	-	-	-	-	<b>78.05</b>
<b>Our method</b>	79.58	87.83	<b>83.21</b>	<b>87.22</b>	73.67

Some of the recognition results with edit distance correction from ICDAR and SVT are shown in Fig. 17(b). As we can observe, our method could recognize scene text with low resolution, different fonts, and distortions.

The reasons that our method achieves a considerable increase in recognition rates on ICDAR03(FULL) and ICDAR03(50) mainly lie in: 1) the part-based tree-structured character detection model makes use of the global structure information and the local appearance information, seamlessly combining character detection, and recognition together and 2) we integrate the detection scores and language model into the character sequence posterior probability from Bayesian decision view so that different types of information could be optimally balanced by maximizing the posterior probability.

Both the character detection and word recognition are implemented in MATLAB. The average processing time to recognize a scene text image is about 3 s on an Intel(R) Core(TM) i7-2600 CPU 3.40-GHZ processor. Since the character detectors are independent from each other, the implementation could be much faster using parallel processing.

#### F. Discussion

1) *Huge Gap*: When we compare the recognition accuracy of the proposed method with and without edit distance correction, we find that the gap is quite large. On the one hand, with edit distance correction, we could achieve a recognition rate of 79.58% with a lexicon of more than 500 words and 87.83% with a lexicon of 50 words on ICDAR11 and ICDAR03, which is quite promising. On the other hand, however, without edit distance correction, the recognition rate is only 58% even when trigram language model is used. We compare the recognition results with and without the correction, and find that most of the inferred results have one or two characters incorrectly recognized. This indicates that the robust reading result without edit distance correction could be further improved if more information could be used or the language model could be more properly incorporated to distinguish the ambiguities between the similar characters. Furthermore, the relatively high recognition accuracy of 79.58% and 87.83% suggests that applications of the word spotting scenario in scene images are possible in the near future.

2) *SVT-More Challenging*: When we compare the recognition accuracy of the proposed method on ICDAR03(50) and

SVT data sets, we find that, without edit distance correction, the recognition rate on ICDAR03 is about 20% higher than that on SVT. Even with edit distance correction, the recognition rate on ICDAR03 is still 15% higher than that on SVT. This suggests that SVT is more challenging than ICDAR03 data set. Since they were collected from Google street view and the images are taken from a moving vehicle, they have a lower resolution than the images in ICDAR03, exhibit more artifacts due to blur, and have larger deformation or distortion. Our character models could detect characters with certain deformation or distortion, as shown in Fig. 17. For characters with large deformation or distortion, as shown in Fig. 16, our character detector might fail. However, some preprocessing methods could be used to correct the deformation or distortion to some extent so that the recognition accuracy could be further improved.

3) *Character Detector*: The character detector is the basis for word recognition. If the detection results contain less false positives and most of the detections are correct, the final recognition results would be quite promising with simple postprocessing strategy. Thus, more attention should be paid to the character detection stage. To this end, we propose to build a part-based TSM for each category of characters so as to make use of both local appearance and global structure information. Although we only use 20–30 images to train each character detector, the performance is quite promising as shown by the experimental results. If we use more training samples to learn the model, the performance could be further improved.

#### VII. CONCLUSION

In this paper, we propose an effective scene text-recognition method incorporating structure-guided detection and linguistic knowledge into the posterior probability of character sequence from Bayesian decision view. We propose to learn a part-based TSM for each category of characters to detect and recognize the characters simultaneously. Since the character-specific TSM makes use of both the global structure information and the local appearance information, the detection results are more reliable. Finally, we get the word-recognition result by maximizing the posterior probability of the character sequence via Viterbi algorithm. We report results on three of the most challenging data sets and the results show that our

method not only outperforms the most popular work published at ICCV 2011 [14] significantly, but also improves the latest results published by Mishra *et al.* [15], [16] considerably. The experimental results show that our method could recognize text in unconstrained scene images with a high accuracy.

## REFERENCES

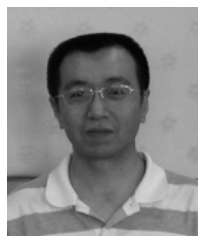
- [1] T. Judd, K. Ehinger, F. Durand, and A. Torralba, "Learning to predict where humans look," in *Proc. IEEE 12th ICCV*, Oct. 2009, pp. 2106–2113.
- [2] S. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, and R. Young, "ICDAR 2003 robust reading competitions," in *Proc. 7th ICDAR*, vol. 2, 2003, pp. 682–687.
- [3] J. Gao and J. Yang, "An adaptive algorithm for text detection from natural scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, vol. 2, Dec. 2001, pp. 1–6.
- [4] X. Chen and A. Yuille, "Detecting and reading text in natural scenes," in *Proc. IEEE CVPR*, vol. 2, Jul. 2004, pp. 366–373.
- [5] M. Lyu, J. Song, and M. Cai, "A comprehensive method for multilingual video text detection, localization, and extraction," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 2, pp. 243–255, Feb. 2005.
- [6] Q. Ye, Q. Huang, W. Gao, and D. Zhao, "Fast and robust text detection in images and video frames," *Image Vis. Comput.*, vol. 23, no. 6, pp. 565–576, Jan. 2005.
- [7] B. Epshtein, E. Ofek, and Y. Wexler, "Detecting text in natural scenes with stroke width transform," in *Proc. IEEE CVPR*, Jun. 2010, pp. 2963–2970.
- [8] L. Neumann and J. Matas, "Real-time scene text localization and recognition," in *Proc. IEEE CVPR*, Dec. 2012, pp. 3538–3545.
- [9] Y. Pan, X. Hou, and C. Liu, "A hybrid approach to detect and localize texts in natural scene images," *IEEE Trans. Image Process.*, vol. 20, no. 3, pp. 800–813, Mar. 2011.
- [10] C. Yi and Y. Tian, "Localizing text in scene images by boundary clustering, stroke segmentation, and string fragment classification," *IEEE Trans. Image Process.*, vol. 21, no. 9, pp. 4256–4268, May 2012.
- [11] P. Shivakumara, T. Phan, and C. Tan, "A Laplacian approach to multi-oriented text detection in video," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 2, pp. 412–419, Feb. 2011.
- [12] C. Shi, C. Wang, B. Xiao, Y. Zhang, and S. Gao, "Scene text detection using graph model built upon maximally stable extremal regions," *Pattern Recognit. Lett.*, vol. 34, no. 2, pp. 107–116, Jan. 2013.
- [13] K. Wang and S. Belongie, "Word spotting in the wild," in *Proc. 11th ECCV*, Sep. 2010, pp. 591–604.
- [14] K. Wang, B. Babenko, and S. Belongie, "End-to-end scene text recognition," in *Proc. ICCV*, Nov. 2011, pp. 1457–1464.
- [15] A. Mishra, K. Alahari, and C. V. Jawahar, "Scene text recognition using higher order language priors," in *Proc. 23rd BMVC*, 2012, pp. 1–11.
- [16] A. Mishra, K. Alahari, and C. V. Jawahar, "Top-down and bottom-up cues for scene text recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Apr. 2012, pp. 2687–2694.
- [17] T. Novikova, O. Barinova, P. Kohli, and V. Lempitsky, "Large-lexicon attribute-consistent text recognition in natural images," in *Proc. 12th ECCV*, Oct. 2012, pp. 752–765.
- [18] J. Weinman, Z. Butler, D. Knoll, and J. Feild, "Toward integrated scene text reading," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 2, pp. 375–387, Jun. 2013.
- [19] J. L. Feild and E. G. Learned-Miller, "Improving open-vocabulary scene text recognition," in *Proc. IEEE ICDAR*, Nov. 2013, pp. 604–608.
- [20] C. Shi, C. Wang, B. Xiao, Y. Zhang, S. Gao, and Z. Zhong, "Scene text recognition using part-based tree-structured character detection," in *Proc. IEEE CVPR*, Jun. 2013, pp. 2961–2968.
- [21] J. Gilavata, R. Ewerth, T. Stefi, and B. Freisleben, "Unsupervised text segmentation using color and wavelet features," in *Image and Video Retrieval*. New York, NY, USA: Springer-Verlag, 2004.
- [22] Y. Song, A. Liu, L. Pang, S. Lin, Y. Zhang, and S. Tang, "A novel image text extraction method based on K-means clustering," in *Proc. 7th IEEE/ACIS ICIS*, May 2008, pp. 185–190.
- [23] D. Chen, J. Olobez, and H. Bourlard, "Text segmentation and recognition in complex background based on Markov random field," in *Proc. 16th Int. Conf. Pattern Recognit.*, vol. 4, 2002, pp. 227–230.
- [24] Q. Ye, W. Gao, and Q. Huang, "Automatic text segmentation from complex background," in *Proc. ICIP*, vol. 5, Oct. 2004, pp. 2905–2908.
- [25] M. Li, M. Bai, C. Wang, and B. Xiao, "Conditional random field for text segmentation from images with complex background," *Pattern Recognit. Lett.*, vol. 31, no. 14, pp. 2295–2308, Oct. 2010.
- [26] C. Shi, B. Xiao, C. Wang, and Y. Zhang, "Adaptive graph cut based binarization of video text images," in *Proc. 10th IAPR Int. Workshop DAS*, May 2012, pp. 58–62.
- [27] P. Shivakumara, T. Phan, S. Bhowmick, C. Tan, and U. Pal, "A novel ring radius transform for video character reconstruction," *Pattern Recognit.*, vol. 46, no. 1, pp. 131–140, 2012.
- [28] J. Feild and E. G. Learned-Miller, "Scene text recognition with bilateral regression," Univ. Massachusetts Amherst, Amherst, MA, USA, Tech. Rep. UM-CS-2012-021, 2012.
- [29] N. Otsu, "A threshold selection method from gray-level histograms," *Automatica*, vol. 11, pp. 285–296, Nov. 1975.
- [30] W. Niblack, *An Introduction to Digital Image Processing*. Mundelein, IL, USA: Strandberg, 1985.
- [31] X. Chen, J. Yang, J. Zhang, and A. Waibel, "Automatic detection and recognition of signs from natural scenes," *IEEE Trans. Image Process.*, vol. 13, no. 1, pp. 87–99, Jan. 2004.
- [32] T. de Campos, B. Babu, and M. Varma, "Character recognition in natural images," in *Proc. Int. Joint Conf. Comput. Vis., Imag. Comput. Graph. Theory Appl.*, Feb. 2009, pp. 1–4.
- [33] A. Newell and L. Griffin, "Multiscale histogram of oriented gradient descriptors for robust character recognition," in *Proc. IEEE ICDAR*, Sep. 2011, pp. 1085–1089.
- [34] A. Coates, B. Carpenter, C. Case, S. Satheesh, B. Suresh, T. Wang, *et al.*, "Text detection and character recognition in scene images with unsupervised feature learning," in *Proc. IEEE ICDAR*, Sep. 2011, pp. 440–445.
- [35] J. Weinman, E. Learned-Miller, and A. Hanson, "Scene text recognition using similarity and a lexicon with sparse belief propagation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 10, pp. 1733–1746, Oct. 2009.
- [36] D. Smith, J. Field, and E. Learned-Miller, "Enforcing similarity constraints with integer programming for better scene text recognition," in *Proc. IEEE CVPR*, Jun. 2011, pp. 73–80.
- [37] A. Berg, T. Berg, and J. Malik, "Shape matching and object recognition using low distortion correspondences," in *Proc. IEEE CVPR*, vol. 1, Jan. 2005, pp. 26–33.
- [38] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 4, pp. 509–522, Apr. 2002.
- [39] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE CVPR*, vol. 1, Jun. 2005, pp. 886–893.
- [40] K. Elagouni, C. Garcia, F. Mamalet, and P. Sébillot, "Combining multi-scale character recognition and linguistic knowledge for natural scene text OCR," in *Proc. 10th IAPR Int. Workshop DAS*, Mar. 2012, pp. 120–124.
- [41] T. Wang, D. J. Wu, A. Coates, and A. Y. Ng, "End-to-end text recognition with convolutional neural networks," in *Proc. 21st ICPR*, Nov. 2012, pp. 3304–3308.
- [42] P. Felzenszwalb and D. Huttenlocher, "Pictorial structures for object recognition," *Int. J. Comput. Vis.*, vol. 61, no. 1, pp. 55–79, 2005.
- [43] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1627–1645, Sep. 2010.
- [44] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. (2008, Oct. 17). *The PASCAL Visual Object Classes Challenge 2008 (VOC2008) Results* [Online]. Available: <http://www.pascal-network.org/challenges/VOC/voc2008/workshop/index.html>
- [45] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. (2009, Oct. 3). *The PASCAL Visual Object Classes Challenge 2009 (VOC2009) Results* [Online]. Available: <http://www.pascal-network.org/challenges/VOC/voc2009/workshop/index.html>
- [46] Y. Yang and D. Ramanan, "Articulated pose estimation with flexible mixtures-of-parts," in *Proc. IEEE CVPR*, Jun. 2011, pp. 1385–1392.
- [47] Z. Xiangxin and D. Ramanan, "Face detection, pose estimation, and landmark localization in the wild," in *Proc. IEEE CVPR*, Jun. 2011, pp. 2879–2886.
- [48] A. Shahab, F. Shafait, and A. Dengel, "ICDAR 2011 robust reading competition challenge 2: Reading text in scene images," in *Proc. IEEE ICDAR*, Sep. 2011, pp. 1491–1496.
- [49] Q. Wang, F. Yin, and C. Liu, "Handwritten Chinese text recognition by integrating multiple contexts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 8, pp. 1469–1481, Aug. 2012.
- [50] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. New York, NY, USA: Wiley, 2012.

- [51] A. Stolcke, "SRILM—An extensible language modeling toolkit," in *Proc. Int. Conf. Spoken Lang. Process.*, vol. 2, 2002, pp. 901–904.
- [52] C. Yi and Y. Tian, "Text extraction from scene images by character appearance and structure modeling," *Comput. Vis. Image Understand.*, vol. 117, no. 2, pp. 182–194, Feb. 2013.
- [53] (2014). *ABBYY FineReader 9.0*. [Online]. Available: <http://www.abbyy.com>



**Cun-Zhao Shi** (S'13) received the B.S. degree in electronic engineering from Wuhan University, Wuhan, China, in 2009. She is currently working toward the Ph.D. degree in pattern recognition and intelligent systems with the Institute of Automation, Chinese Academy of Sciences, Beijing, China.

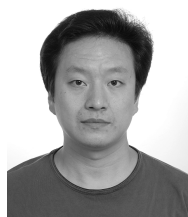
Her research interests include pattern recognition, computer vision, text detection, text extraction, character recognition, and text recognition in natural images.



**Chun-Heng Wang** (M'10) received the B.S. and M.E. degrees in electronic engineering from Dalian University of Technology, Dalian, China, and the Ph.D. degree in pattern recognition and intelligent control from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 1993, 1996, and 1999, respectively.

He has been a Professor with State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, since 2004, where he is currently a

Professor. His research interests include pattern recognition, image processing, neural networks, and machine learning.



**Bai-Hua Xiao** (M'10) received the B.S. degree in electronic engineering from Northwestern Polytechnical University, Xi'an, China, and the Ph.D. degree in pattern recognition and intelligent control from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 1995 and 2000, respectively.

He has been a Professor with State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, since 2005, where he is currently a

Professor. His research interests include pattern recognition, computer vision, image processing, and machine learning.



**Song Gao** received the B.S. degree in electronic engineering from Xi'an Jiaotong University, Xi'an, China, in 2010. He is currently working toward the Ph.D. degree in pattern recognition and intelligent systems with the Institute of Automation, Chinese Academy of Sciences, Beijing, China.

His research interests include pattern recognition, text detection, character recognition, and text recognition in natural images.



**Jin-Long Hu** received the B.S. degree in electronic engineering from Xi'an Jiaotong University, Xi'an, China, in 2010. He is currently working toward the Ph.D. degree in pattern recognition and intelligent systems with the Institute of Automation, Chinese Academy of Sciences, Beijing, China.

His research interests include pattern recognition, text detection, extraction, and recognition in natural images.