Effectively Leveraging Visual Context to Detect Texts in Natural Scenes

Jiyan Pan, Ye Chen, Bo Anderson, Pavel Berkhin, Takeo Kanade

Carnegie Mellon University, Microsoft Corporation

Abstract. Detecting texts in natural scenes is challenging because of large variation in size and layout of texts and strong distractions from background clutters. Leveraging contextual information is crucial in boosting the detection accuracy. In this paper, we construct a conditional random field (CRF) to utilize visual context that helps enhance true detections and suppress false alarms. Unlike previous works, the pairwise potentials in our model encode three different compatibility/repulsion relationships among character candidates under two different layout scenarios, and the unary potentials are obtained from multi-class recognition confidence of individual character candidates. In addition, we use easy texts to help recover difficult ones in an iterative manner. Due to these efforts, our method outperforms state-of-the-art text detection algorithms on the challenging ICDAR dataset.

1 Introduction

Text detection in natural scenes has a wide range of applications, including augmented reality, image retrieval, and robotic navigation, etc. Extracting texts from natural scenes, however, is much more difficult than reading off texts from scanned materials. Several examples are shown in Figure 1, where texts could appear anywhere in the image with different sizes and layouts. Also, background clutters frequently cause distractions. As a result, directly applying commercial optical character recognition (OCR) engines to natural scenes usually gives poor performance [1].

A major factor underlying the challenges of detecting texts in natural scenes is that there usually exist many text-like components in a natural scene which, when viewed in isolation, are hardly distinguishable from texts. Therefore, context plays an important role in disambiguating candidate characters. There are two types of contexts here: linguistic context and visual context. The former one resorts to a language model to eliminate invalid character sequences [2]. The latter one utilizes visual information from other parts of the image to help determine if a candidate character is indeed text. This paper focuses on how to effectively leverage *visual* context.

Existing works that resort to visual context can be roughly categorized into texture-based and connected-component-based (CC-based) methods. Texture-based methods, which rely on collective features of text regions, implicitly utilize context by considering multiple characters simultaneously when extracting

Fig. 1. Examples of detecting texts in natural scenes. Left column: original images. Right column: detection results of our algorithm. The red, blue and green boxes are group, word and character bounding boxes, respectively. The yellow letters are the recognition results obtained during detection. Best viewed in color.

features [3–12]. Although this type of approaches are more reliable than relying on individual characters alone, they usually have a hidden requirement that the distributions of text densities in testing images be similar to those in training images. Consequently, they are not effective when novel spacings and/or sizes are encountered during testing.

Instead of classifying image regions, CC-based methods propose connected components in the image as candidate characters, and determine the "characterness" for each candidate [13–16]. As the information provided by individual candidates themselves is usually insufficient to evaluate their characterness, visual context is explicitly exploited by checking visual consistency among nearby candidate characters. For example, Yi *et al.* designed a set of rules to make sure valid text lines contain candidates of similar visual properties [13]. Conditional Random Field (CRF) [17] is employed in [14–16] to assign optimal text/background labels to candidates. Visual context is explicitly encoded in the pairwise potentials of the CRF models. This type of approaches are less vulnerable to the variation in text density than texture-based methods, and therefore achieve the state-of-the-art performance.

The algorithm we propose in this paper falls in the CC-based category. We also use a CRF to leverage visual contextual information. However, there are several major distinctions. Firstly, when encoding visual context in pairwise potentials, we carefully model three different types of character relationships and their implications, instead of simply encouraging smooth labeling among neighboring nodes as is done in [15]. Also, depending on whether bounding boxes overlap, we use different features when computing pairwise potentials, instead of mixing the two scenarios as is done in [16, 14]. Secondly, confidence of multi-class optical character recognition (OCR) is incorporated in the unary potential of our CRF model. This differs from [15] in which binary text/background classification scores are used as unary potentials. As our algorithm utilizes the knowledge of the appearance of the entire character set, it is more effective than performing a gross binary classification. Our algorithm also differs from Zhang *et al.* who apply OCR in a separate filtering stage rather than incorporating OCR scores

into the CRF model [14]. Thirdly, we use easy texts to help recover difficult ones iteratively. As far as our knowledge is concerned, this strategy has yet to be reported in related literature.

Attributed to the factors mentioned above, we are able to leverage visual context in a more effective manner, outperforming the state-of-the-art CRF based algorithms by 2 to 14% on the ICDAR dataset [18]. Our algorithm also achieves the best performance among all the text detection algorithms presented in the ICDAR text locating competitions [19, 20].

The rest of this paper is organized as follows. Our algorithm is described in detail in Section 2, followed by experimental results presented in Section 3, and we conclude this paper in Section 4.

2 Algorithm description

The overall scheme of our algorithm is summarized in Figure 2. Individual components in the flowchart will be detailed in the subsections below.

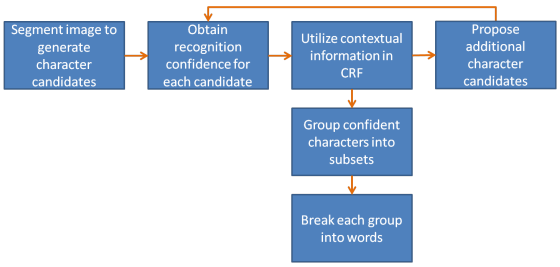

Fig. 2. The flowchart of our approach.

2.1 Initialize character candidates

We initialize character candidates from the segments given by stroke width transform (SWT) [1], as we found that SWT-generated segments correspond to true characters better than other segmentation methods. The core idea of SWT is that texts usually have roughly constant stroke widths. Therefore, SWT estimates the stroke width at each pixel location, and pixels with similar stroke widths are grouped together to form segments which serve as character candidates. An example is shown in Figure 3. The center image contains brighter segments surrounded by darker regions, while the rightmost image contains darker segments surrounded by brighter regions. We could see that lots of background clutters are also included into the set of character candidates. From this point on, our algorithm significantly differs from [1]: we utilize character recognition and contextual inference in lieu of a set of heuristics.

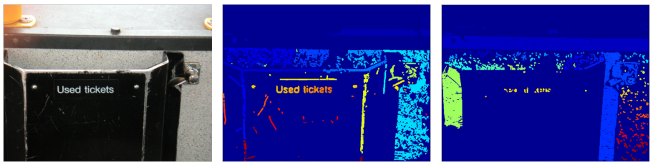

Fig. 3. SWT-generated segments. The leftmost image is the original image. The center image shows bright-center-dark-surround segments, where different colors indicate different segments, except for the large dark-blue region which indicates background and is not considered. The rightmost image contains dark-center-bright-surround segments. Best viewed in color.

2.2 Obtain recognition confidence

Instead of applying binary classification to distinguish characters from background, we perform multi-class character recognition on each candidate and use recognition confidence as an initial measure of how likely a candidate is a character (*i.e.* characterness). In other words, only when a candidate resembles one of the learned characters can it be regarded as text.

To alleviate the problem that several characters might get connected due to the imperfection of segmentation, we divide each segment into several smaller segments according to its aspect ratio, and those new segments are added to the character candidate pool as well.

Each candidate is fed into a multi-class Random Forest (RF) classifier [21, 22] that attempts to classify the candidate into one of the 62 categories. Those categories include all upper and lower-case letters as well as 10 digits

After the predicted category is obtained from the RF classifier, we proceed to evaluate the recognition confidence. For multi-way classifiers, the entropy of label distribution output by the classifier is commonly used as an indicator of classification confidence: lower entropy (i.e. more peaky distribution) indicates higher confidence [21]. However, it is often observed that a false character candidate might resemble none of the trained character classes, yet it is still, relatively speaking, far closer to one class than the rest of the classes. In this case, the entropy would still be low, erroneously indicating a high confidence. Therefore, we do not resort to the class distribution entropy generated by the RF classifier to evaluate recognition confidence. Instead, we compute the distances in feature space between the candidate and all the training examples *within the predicted category*. The minimal distance is converted to the recognition confidence by a sigmoid function:

$$c_i = \frac{1}{1 + \exp(\lambda_U (d_i - d_0))}, \tag{1}$$

where c_i and d_i are the recognition confidence and the minimal distance for candidate i, respectively; λ_U and d_0 are the parameters that control the shape of the conversion curve. Their optimal values are determined by maximizing the performance on the ICDAR training set [18].

The character recognizer is trained on the "EnglishFnt" subset of the Chars74K dataset [23]. Completely independent of the ICDAR dataset, the Chars74K

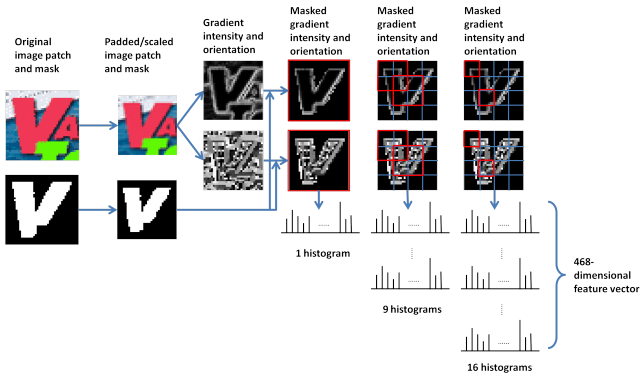

Fig. 4. Extracting hierarchical HOG features from a character candidate. The original image patch and mask are padded and scaled to a canonical size. Gradient is signed and each histogram is a 18-dimensional vector. Best viewed in color.

dataset contains computer generated characters of various fonts. The reason for using this dataset instead of the ICDAR training set is that a) not all the 62 characters appear in the ICDAR training set, b) the ground-truth bounding boxes of the characters in the ICDAR training set do not fit well with the characters, and c) we would like to see if a recognizer trained on synthetic characters could generalize well in natural scenes.

The features we use for character recognition are three-level hierarchical Histogram of Oriented Gradients (HOG) [24]. The feature extraction process is illustrated in Figure 4. Note that during training, no mask is needed as each training example in the Chars74K dataset has a clear background; during testing, the mask for a character candidate is the segment mask.

The recognition confidence serves as the initial detection score (*i.e.* characterness). Several examples are shown in the center column of Figure 5. As we can see, although many true characters have relatively high recognition confidence, some of them still have low confidence due to incorrect segmentation and/or interference from other objects. In addition, many background clutters have high recognition confidence because of their resemblance to characters when observed in isolation. This is inevitable because characters usually do not carry sufficient discriminative information against many background structures. In the next sub-section, we describe how we leverage visual context to improve detection performance.

2.3 Utilize visual context

The intuition of enhancing detection performance by utilizing visual context is that, if two nearby candidates have similar color, size, and stroke width, then it is more likely that they are both characters. On the other hand, if the bounding boxes of the two candidates significantly overlap, then it is unlikely that they are *both* characters. Therefore, a candidate with high confidence could help promote

Fig. 5. Our algorithm uses visual context to enhance true detections and suppress false detections. The leftmost column contains the original images. The center column shows the detection score (a.k.a. "characterness") obtained from applying a character recognizer to individual character candidates separately. Here, brighter red indicates higher characterness. The rightmost column displays the refined characterness after CRF inference that incorporates contextual information. Best viewed in color.

the confidence of its nearby candidates having similar properties, while suppressing the confidence of those having mutually exclusive positions. Of course by doing so we run the risk of promoting the confidence of false detections that happen to share similar properties with nearby true detections, yet fortunately this rarely happens in practice.

We use a Conditional Random Field (CRF) [17] to achieve this purpose. We define each character candidate i to be a node n_i in the CRF, and there is an edge between two candidates if they are *both* within the influence field of the other. The influence field of a candidate is illustrated by the shaded yellow region in Figure 6. The extent of the influence field determines how densely connected the CRF is. In our experiments, the influence field is set to be four times the width of the candidate on each side, and has a tilt angle of $\pi/9$.

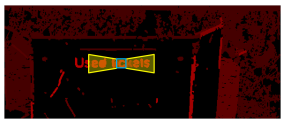

Fig. 6. The influence field of a character candidate, where the character candidate is enclosed by the blue box, and the influence field is the shaded yellow region (including the region within the blue box). Best viewed in color.

The state, S_i, of each node can take on two values: 0 meaning non-character, and 1 meaning character. The goal is to find an optimal joint state assignment, $\hat{\mathbf{S}}$, for all the nodes, given image features X:

$$\hat{\mathbf{S}} = \arg\max_{\mathbf{S}} \log P(\mathbf{S}|X), \tag{2}$$

where $\mathbf{S}$ is a joint state assignment of all the nodes, and $\log P(\mathbf{S}|X)$ is the log conditional distribution of joint state assignment given image features. This term could be decomposed into unary and pairwise potentials as follows:

$$\log P(\mathbf{S}|X) = \eta_u \sum_i \omega_i(S_i|X_i) + \eta_p \sum_{i,j} \phi_{ij}(S_i, S_j|X_{ij}) - \log Z(X), \tag{3}$$

where $\omega_i(S_i|X_i)$ is the unary potential for candidate i, $\phi_{ij}(S_i, S_j|X_{ij})$ is the pairwise potential for candidates i and j, $Z(X)$ is the partition function, and η_u and η_p are the weights for the unary and pairwise potentials, respectively. The weights are determined by maximizing the detection F-measure on the ICDAR training set.

In our model, the unary potential for node i is defined based on the multi-class recognition confidence c_i obtained from Equation 1:

$$\omega_i(S_i|X_i) = \begin{cases} 1 - c_i & \text{if } S_i = 0 \\ c_i & \text{if } S_i = 1 \end{cases} \tag{4}$$

To derive the pairwise potential, we first categorize pairwise relationships into three types:

Compatible: the bounding boxes of two candidates do not overlap and the two candidates have similar properties;
Irrelevant: the bounding boxes of two candidates do not overlap and the two candidates have dissimilar properties;
Repulsive: the bounding boxes of two candidates overlap.

To evaluate the degrees of the three pairwise relationships, we compute two scores. The first score is *similarity score*, V_S. This score measures the degree of compatibility between two candidates. Its value ranges from 0 to 1, where 0 indicates an irrelevant pairwise relationship and 1 indicates a highly compatible one. The score consists of color, size, and stroke width similarity between the two candidates:

$$V_S = \exp\{-\lambda_S(\frac{\|\bar{\mathbf{g}}_i - \bar{\mathbf{g}}_j\|^2}{2\sigma_C^2} + \frac{\max(0, r_0 - r_{ij})^2}{2\sigma_S^2} + \frac{(w_i - w_j)^2}{2\sigma_W^2})\}, \tag{5}$$

where $\bar{\mathbf{g}}_i$ and $\bar{\mathbf{g}}_j$ are the average colors of candidates i and j, respectively; $r_{ij} = \min(h_i, h_j)/\max(h_i, h_j)$ is the relative height ratio of the two candidates; r_0 is the lower limit of the height ratio that does not incur penalty, and is set to be 0.6 to account for height differences among letters; w_i and w_j are the stroke widths of candidates i and j, respectively; λ_S, σ_C, σ_S, and σ_W control the sensitivity of the similarity score. Their reasonable values are determined from the ICDAR training set.

The second score is *repulsion score*, V_R. This score measures the degree of repulsion between two candidates. Its value also ranges from 0 to 1, where 0 indicates an irrelevant pairwise relationship and 1 indicates a highly repulsive

one. The repulsive score is derived from the degree of overlap of the bounding boxes of the two candidates:

$$V_R = 1 - \min\left(1, \frac{\|\mathbf{x}_i - \mathbf{x}_j\|}{\max\left(D_i, D_j\right)}\right)^{\lambda_R}, \tag{6}$$

where $\mathbf{x}_i$ and $\mathbf{x}_j$ are the center coordinates of the two bounding boxes, D_i and D_j are the half diagonals of the two bounding boxes, and λ_R controls the sensitivity of the repulsion score. It's reasonable value is also determined from the ICDAR training set.

	Both are characters	One character, one non-character	Both are non-characters
Compatible	Likely	Unlikely	Neutral
Irrelevant	Neutral	Neutral	Neutral
Repulsive	Unlikely	Likely	Likely

Fig. 7. Implications of the three types of pairwise relationships in terms of characterness. For example, being compatible implies it is unlikely that one is character and the other is non-character.

Now we look at the implications of the three types of pairwise relationships in terms of characterness. The implications are summarized in Figure 7. In order to encode them, we define the pairwise potential in such a way that its value for "most likely", "neutral", and "most unlikely" is 1, 0.5, and 0, respectively:

* **If the bounding boxes of the two candidates do not overlap** (*i.e.* encoding the characterness implications for the pairwise relationships from being highly compatible to being irrelevant),

$$\phi_{ij}(S_i, S_j|X_{ij}) = \begin{cases} 0.5(1+V_S) & \text{if } S_i = S_j = 1 \\ 0.5 & \text{if } S_i = S_j = 0 \\ 0.5(1-V_S) & \text{else} \end{cases} \tag{7}$$

* **If the bounding boxes of the two candidates overlap** (*i.e.* encoding the characterness implications for the pairwise relationships from being highly repulsive to being irrelevant),

$$\phi_{ij}(S_i, S_j|X_{ij}) = \begin{cases} 0.5(1-V_R) & \text{if } S_i = S_j = 1 \\ 0.5(1+V_R) & \text{else} \end{cases} \tag{8}$$

Note that our model does not just smooth the labeling; it also encourages nearby nodes to take different labels if visual evidence suggests a repulsive relationship. Also, overlapping and non-overlapping bounding boxes are treated differently when we compute the pairwise potential. This allows for a better characterization of interactions among adjacent candidates under different scenarios.

After all the potentials are obtained, we run max-product belief propagation [25] over the CRF to infer the optimal joint state assignment to each candidate. The inference also gives each candidate a posterior probability of being a character (*i.e.* a refined characterness).

Sometimes a false detection with high confidence would have irrelevant relationships with most of its nearby candidates, and therefore receives neither support nor suppression from its neighbors. To handle this type of false detections, we forcibly set their refined characterness to be zero if a candidate does not receive any support from its neighbors. Of course, standalone characters would be mistakenly suppressed in this process, yet in most applications only words and sentences containing multiple characters are of interest.

The refined characterness of several example images are shown in the rightmost column of Figure 5. We could see that many background regions are suppressed, including those virtually indistinguishable from characters if observed in isolation. In the meantime, the characterness of some true characters that are originally not so confident are enhanced by nearby confident candidates.

After the refined characterness are obtained, all the candidates whose characterness are above a threshold τ are regarded as detected characters.

2.4 Propose additional character candidates

In some cases, the SWT segmentation algorithm might fail to segment out a true character in the first place and/or over-segment a character. When this happens, the CRF inference has no way to correctly recover the whole character. In order to address this problem, we use the already-detected characters to "illuminate" their surrounding regions and propose additional character candidates that might hopefully include those missing ones. (Please see Figure 8.)

To achieve this purpose, we use the color statistics of those detected characters to classify the pixels in their influence fields into either text or background. More specifically, for each detected character, we summarize the color statistics of the character by fitting three Gaussians to the three color components of all the pixels on the character. To represent the color statistics of the background region around the character, we also fit three Gaussians to the three color components of all the pixels within the image regions immediately above and below the character's bounding box. Those six Gaussians (three for each class) form a naive Bayes classifier associated with the character. The class prior is assumed to be flat.

After we have learned the naive Bayes classifiers for all the detected characters, we classify each pixel in the image as follows. If a pixel is not within the influence field of any detected characters, the pixel is classified as background. Otherwise, classify the pixel using the naive Bayes classifier associated with each detected character *that has influence over the pixel*. A detected character would regard the pixel as text if its associated naive Bayes classifier labels the pixel as text *and* the color of the pixel is close enough to that of the detected character (*i.e.*, within the 3σ confidence interval of the Gaussians describing the character). The pixel is finally classified as text if at least one of the detected characters regards it as text.

We obtain a binary text map after all the pixels in the image have been classified. Additional character candidates are proposed by obtaining the connected components of the text map, and are added into the original candidate pool

generated by SWT. Also, we remove from the pool those candidates located completely within the non-text region of the text map. Then we run character recognition and CRF inference again. This procedure could iterate multiple times, yet in practice we found two iterations are sufficient.

An example is shown in Figure 8. Due to the imperfection of SWT-based segmentation, some letters are broken into disjoint segments (see Figure 8(b)). Consequently, those letters are not correctly detected (see Figure 8(c)). After using the color statistics of nearby detected texts, those missing segments are "illuminated", and good segments corresponding to whole letters are obtained from the connected components in the text map (see Figure 8(d)). Note that although a lot of background clutters are also included, they are subsequently rejected in the character recognition and CRF inference processes, yielding a much better result in Figure 8(e) than in (c).

Fig. 8. Missing text strokes are recovered by using the color information from already-detected texts. (a) Original image. (b) Segments from SWT. Note that some letters are over-segmented. (c) Characterness after the first round of CRF inference. (d) Text map obtained by using the color statistics of the detected characters in the first round. The influence field of each detected character is enlarged to be 20 times the character width. Connected components in the text map are proposed as additional character candidates. (e) Characterness after the second round of CRF inference. (f) Final detected texts. Best viewed in color.

2.5 Form text groups and words

In order to further extract words from detected characters, we first group the characters into subsets according to the pairwise compatibility among them. The pairwise compatibility is readily available from pairwise potentials $\phi_{ij}(S_i, S_j|X_{ij})$. We form a distance matrix between each pair of characters where the distance between characters i and j is defined as

$$d_{ij} = \begin{cases} 1 - \phi_{ij}(S_i, S_j|X_{ij}) & \text{if } e_{ij} = 1 \\ \infty & \text{if } e_{ij} = 0 \end{cases} \tag{9}$$

where, $e_{ij} = 1$ means there is an edge between characters i and j in the CRF network. After obtaining the distance matrix, we perform single-linkage hierarchical clustering over the characters. The resulting groups are characters of similar color, size, and stroke width located roughly on the same line. Please refer to Figures 1 and 10 for examples.

As each group might contain more than one word, we break each group into words according to the spacing between adjacent characters. More specifically, for a character i, denote the x-coordinate of the left side, center, and right side of its bounding box as x_i^l, x_i^c, and x_i^r, respectively. Suppose character j is immediately to the right of character i, then our algorithm declares a blank space between them if $[(x_j^l - x_i^r) - median_{mn}(x_n^l - x_m^r)]/mean_{mn}(x_n^c - x_m^c) > \tau_w$, where $median_{mn}(x_n^l - x_m^r)$ and $mean_{mn}(x_n^c - x_m^c)$ denote the median boundary gap and average center spacing between all adjacent characters in the group. τ_w is a threshold controlling the sensitivity of group breaking. Its optimal value is determined from the ICDAR training set.

3 Experiments

We conduct experiments on the ICDAR dataset [18]. The dataset contains 258 images for training and 251 images for testing. Ground truth is provided in the form of a bounding box for each word. To compare the outputs of our algorithm against the ground truth, we put a bounding box around each detected word and compute precision and recall on word level according to the specifications of the ICDAR text detection competitions [19, 20].

Algorithm	Precision	Recall	F-measure
Ours (full+filter)	**0.73**	**0.68**	**0.71**
Ours (full)	**0.71**	**0.68**	**0.70**
Ours (baseline+CRF)	0.73	0.65	0.69
Ours (baseline)	0.70	0.51	0.59
Pan et al. [16]	0.67	0.71	0.69
Epshtein et al. [1]	0.73	0.60	0.66
Yi et al.[13]	0.71	0.62	0.66
Neumann et al. [2]	0.59	0.55	0.57
Zhang et al.[14]	0.57	0.57	0.57
H. Becker	0.62	0.67	0.62
A. Chen	0.60	0.60	0.58
Q. Zhu	0.33	0.40	0.33
J. Kim	0.22	0.28	0.22
N. Ezaki	0.18	0.36	0.22
Ashida	0.55	0.46	0.50
HWDavid	0.44	0.46	0.45
Wolf	0.30	0.44	0.35
Todoran	0.19	0.18	0.18
Full	0.1	0.06	0.08

Fig. 9. Performance comparison of different algorithms. The top four rows are different versions of our algorithm. The next five rows show the performance of several state-of-the-art algorithms, and the remaining are the participating algorithms in the ICDAR competitions [20].

The algorithms we compare with include several state-of-the-art text detection algorithms, as well as the participating algorithms in the ICDAR competitions. In order to evaluate the contribution of the CRF inference component described in Section 2.3 and the missed candidate retrieval component described in Section 2.4, we also perform experiments when both the two components are removed (*i.e.* baseline) and when only the missed candidate retrieval component is removed (*i.e.* baseline+CRF). For each of the three versions of our algorithm, the detection threshold τ is separately determined by maximizing the F-measure on the ICDAR training set.

The quantitative results are summarized in Figure 9. We could see that when text detection is relying solely on individual character detection without utilizing any context, the performance is relatively poor, as is shown in the row "Ours (baseline)". However, when we leverage contextual information by adding the CRF inference component, there is a significant boost in performance, as is shown in the row "Ours (baseline+CRF)". Further using detected characters to recover missing characters yields additional improvement that pushes the F-measure to 0.70. We achieve even better performance in the row "Ours (full+filter)"after applying a simple filter that discards any detected group in which all the characters are either 'I' or '1' or 'l'. This filter helps remove false detections resulting from highly regular 'I'-shaped repeated background structures that enhance each other during the CRF inference.

Our algorithm outperforms state-of-the-art approaches listed in the middle section of the table in Figure 9. All of those approaches are CC-based algorithms, and some of them (*e.g.* [16] and [14]) also employ a CRF to utilize contextual information. Note that even without the missed candidate retrieval component (*i.e.* Ours(baseline+CRF)), the performance of our algorithm is already among the highest. Compared to some of the state-of-the-art approaches, an extra benefit of our algorithm is that character recognition results are obtained along the way, without the need for a separate OCR stage.

Qualitative results of our algorithm are displayed in Figure 10. We observe that although background clutters contain many text-like segments (*e.g.* window frames, ceiling structure, foliage, *etc.*), our algorithm is able to effectively suppress them after leveraging the visual context using the CRF model we have proposed. We also observe that some hard-to-detect characters (*e.g.* several blurred characters in the lower-right image of Figure 10) are effectively recovered by our iterative procedure that retrieves missed candidates. Also note that although the testing images contain texts of various densities. our algorithm is not affected at all because it does not rely on collective region features.

Although not part of the text detection task, recognition results are also included in Figures 10, where red letters indicate recognition mistakes. Although some characters are not correctly recognized, most of them still have correct bounding boxes. Therefore, a more advanced recognition technique could further improve the recognition performance.

Figure 11 shows several cases when our algorithm fails to detect texts. Many of the missed texts in those images either do not have sufficient contrast against the background, or are too small to detect.

A weakness of our approach is that characters not trained upon might not be correctly detected and/or recognized. However, this problem could be alleviated simply by training the system on more characters for various languages.

We implemented our algorithm in Matlab on a 3.47GHz desktop. It takes about 30 to 90 seconds to process an image, depending on its size.

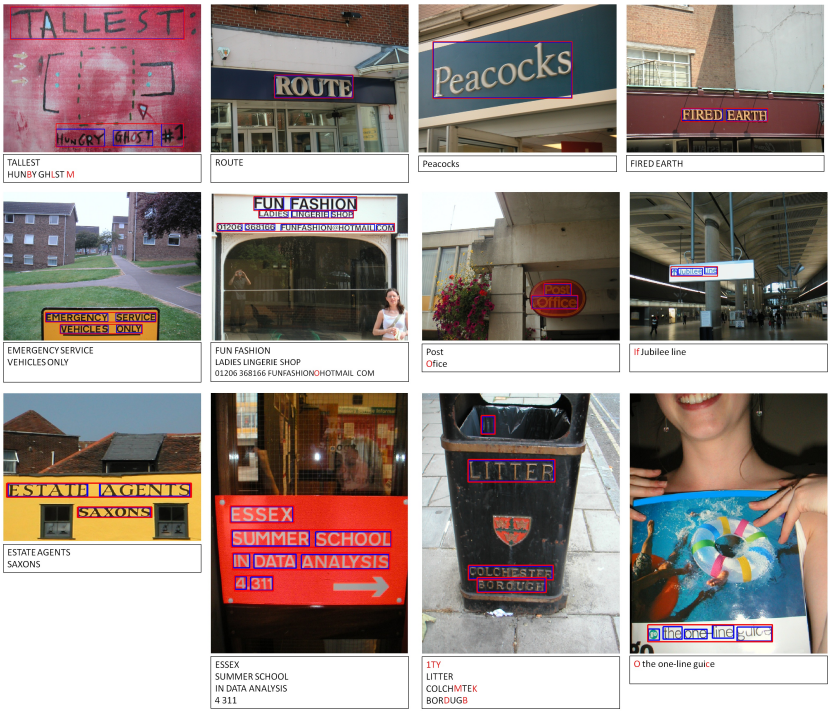

Fig. 10. Detection results on testing images. The red and blue rectangles represent group and word bounding boxes, respectively. We do not show character bounding boxes for clarity. Recognition results are shown below each image, where letters in red indicate mistakes. No separate OCR engine is used. Best viewed in color.

Fig. 11. Cases when our algorithm fails to detect texts.

4 Conclusion

We presented a text detection algorithm that effectively leverages visual context by applying a CRF model in which unary potentials are derived from multi-class

recognition confidence and pairwise potentials could capture various types of interactions among nearby character candidates. We also use the color statistics of easier characters to recover harder ones. As a result, our algorithm achieves better performance than the state of the art on the challenging ICDAR dataset.

References

1. Epshtein, B., Ofek, E., Wexler, Y.: Detecting text in natural scenes with stroke width transform. CVPR (2010)
2. Neumann, L., Matas, J.: A method for text localization and recognition in real-world images. ACCV (2010)
3. Chen, X., Yuille, A.L.: Detecting and reading text in natural scenes. CVPR (2004)
4. Hanif, S.M., Prevost, L., Negri, P.A.: A cascade detector for text detection in natural scene images. ICPR (2008)
5. Hanif, S.M., Prevost, L.: Text detection and localization in complex scene images using constrained adaboost algorithm. ICDAR (2009)
6. Liu, C., Wang, C., Dai, R.: Text detection in images based on unsupervised classification of edge-based features. Document Analysis and Recognition (2005)
7. Liu, X., Samarabandu, J.: Multiscale edge-based text extraction from complex images. ICME (2006)
8. Liu, Z., Sarkar, S.: Robust outdoor text detection using text intensity and shape features. ICPR (2008)
9. Pan, W., Bui, T.D., Suen, C.Y.: Text detection from scene images using sparse representation. ICPR (2008)
10. Shivakumara, P., Phan, T.Q., Tan, C.L.: A gradient difference based technique for video text detection. ICDAR (2009)
11. Shivakumara, P., Phan, T.Q., Tan, C.L.: A laplacian approach to multi-oriented text detection in video. PAMI (2011)
12. Tran, H., Lux, A., Nguyen, H., Boucher, A.: A novel approach for text detection in images using structural features. ICAPR (2005)
13. Yi, C., Tian, Y.: Text string detection from natural scenes by structure-based partition and grouping. TIP (2011)
14. Zhang, H., et al.: An improved scene text extraction method using conditional random field and optical character recognition. ICDAR (2011)
15. Peng, X., H. Cao, R.P., Natarajan, P.: Text extraction from video using conditional random fields. ICDAR (2011)
16. Pan, Y.F., Hou, X., Liu, C.L.: Text localization in natural scene images based on conditional random field. ICDAR (2009)
17. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: probabilistic models for segmenting and labeling sequence data. ICML (2001)
18. : (Icdar dataset) http://algoval.essex.ac.uk/icdar/Datasets.html.
19. : Icdar 2003 robust reading competitions. ICDAR (2003)
20. : Icdar 2005 robust reading competitions. ICDAR (2005)
21. Breiman, L.: Random forests. Machine Learning **45** (2001) 5–32
22. : (Random forest codes) http://www.stat.berkeley.edu/ breiman/RandomForests/.
23. : (The chars74k dataset) http://www.ee.surrey.ac.uk/CVSSP/demos/chars74k/.
24. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. CVPR (2005)
25. Heskes, T.: Stable fixed points of loopy belief propagation are minima of the bethe free energy. Advances in NIPS (2003)