Scene Text Recognition: No Country for Old Men?

Lluís Gómez(✉) and Dimosthenis Karatzas

Computer Vision Center, Universitat Autònoma de Barcelona, Barcelona, Spain
{lgomez,dimos}@cvc.uab.es

Abstract. It is a generally accepted fact that Off-the-shelf OCR engines do not perform well in unconstrained scenarios like natural scene imagery, where text appears among the clutter of the scene. However, recent research demonstrates that a conventional shape-based OCR engine would be able to produce competitive results in the end-to-end scene text recognition task when provided with a conveniently preprocessed image. In this paper we confirm this finding with a set of experiments where two off-the-shelf OCR engines are combined with an open implementation of a state-of-the-art scene text detection framework. The obtained results demonstrate that in such pipeline, conventional OCR solutions still perform competitively compared to other solutions specifically designed for scene text recognition.

1 Introduction

The computer vision research community has dedicated a significant research effort on text extraction systems for text in natural scene images over the last decade. As a result, scene text extraction methods have evolved substantially and their accuracy has increased drastically in recent years [6], see the evolution of detection algorithms in the ICDAR competitions in Fig. 1. However, the problem is still far from being considered solved: note that the winner methods in the last ICDAR competition achieve only 66 and 74 % recall in the tasks of text localization and text segmentation respectively, while the best scoring method in cropped word recognition achieved a 83 % recognition rate. The main difficulties of the problem stem from the extremely high variability of scene text in terms of scale, rotation, location, physical appearance, and typeface design. Moreover, text in scene images may suffer from several degradations like blur, illumination artifacts, or may appear in extremely low quality.

Most of the published methods on Robust Reading systems focus on specific problems of the end-to-end pipeline, i.e. detection [3,15], extraction [5], and recognition [14] are traditionally treated as separate problems. This is quite natural if one takes into account that earlier works on text detection were hardly able to do detection with acceptable rates, see e.g. the 38 % f-score of the winner method in the first ICDAR competition [7], to furthermore think on approaching the full end-to-end problem. More recently, encouraged by the increased accuracy rates in detection and segmentation, end-to-end recognition methods seem viable to produce decent results.

C.V. Jawahar and S. Shan (Eds.): ACCV 2014 Workshops, Part II, LNCS 9009, pp. 157–168, 2015.
DOI: 10.1007/978-3-319-16631-5_12

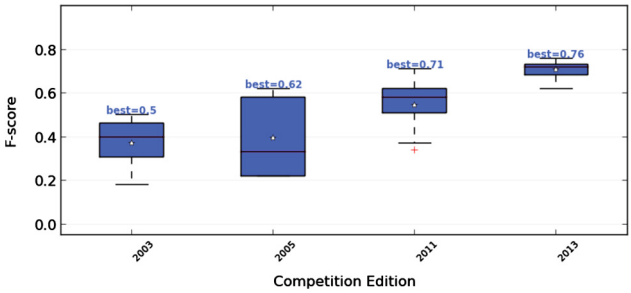

Fig. 1. Evolution of text detection f-score results of the participant methods in the ICDAR Robust Reading competitions. Notice that although dataset may have changed among different editions, results are still reasonably comparable.

In this paper we are interested in end-to-end unconstrained scene text recognition, i.e. in methods that do not make use of small fixed lexicons for the recognition. The firsts works approaching this complex task were proposed more than ten years ago [2], but papers reporting results in public benchmarks date from just few years ago [9,20]. Nowadays there exist reliable applications based on such technology that are already in the hands of millions[1] [1]. However, such real world applications are limited to very well delivered conditions, e.g. horizontally oriented and well focussed bilevel text, and often rely on large-scale data center infrastructure. Thus, there is still room for research on more robust and efficient methods.

A typical experiment frequently repeated to demonstrate the need of specific techniques for scene text detection and recognition is to attempt to process a raw scene image with a conventional OCR engine. This normally produces a bunch of garbage on the recognition output. Obviously this is not the task for which OCR engines have been designed and the recognition may be much better if we provide it with a pixel level segmentation of the text. Figure 2 show the output of the open source Tesseract[2] [17] OCR engine for a raw scene image and for its binarized text mask obtained with a scene text extraction method.

Through this paper we experimentally demonstrate that in such pipeline off-the-shelf OCR engines yield competitive results to state-of-the-art solutions specifically designed for scene text recognition.

While being this primarily an engineering work, we think there are two important aspects of it that can be of broad interest from a research perspective: one is about the question of whether pixel-level text segmentation is really useful for the final recognition, the other is about how stronger language models affect the final results.

[1] Word Lens and the Google Translate service are examples of a real applications of end-to-end scene text detection and recognition that have acquired market-level maturity.

[2] http://code.google.com/p/tesseract-ocr/.

In the following sections we make a comprehensive review of related work, describe the set-up of our end-to-end pipeline, and show the results of our experiments. Finally we close the paper with a valuable discussion.

2 Related Work

Table 1 summarize several aspects of language models used in existing end-to-end approaches for unconstrained recognition, and a more detailed description of such methods is provided afterwards. Table 1 compares the use of character/word n-grams (as well as their sizes), over-segmentation, and dictionaries (here the "Soft" keyword means the method allow Out-Of-Dictionary word recognition, while "Hard" means only "In-Dictionary" words are recognized).

Table 1. Summary of different language model aspects of end-to-end scene text recognition methods.

Method	Dictionary	Char n-gram	Word n-gram	Over-segmentation
Neumann *et al.* [9–12]	Soft (10k)	bi-gram	No	Yes
Wang *et al.* [21]	Hard (hunspell)	No	No	Yes
Neumann *et al.* [13]	No	3-gram	No	Yes
Yao *et al.* [22]	Soft (100k)	bi-gram	No	No
Bissacco *et al.* [1]	Soft (100k)	8-gram	4-gram	Yes

In [9] Neumann and Matas propose the classification of Maximally Stable Extremal Regions (MSERs) as characters and non-characters, and then the grouping of character candidates into text lines with multiple (mutually exclusive) hypotheses. For the recognition each MSER region mask is normalized and resized to a fixed size in order to extract a feature vector based on pixel directions along the chain-code of its perimeter. Character recognition is done with a SVM classifier trained with synthetic examples. Ambiguous recognition of upper-case

Fig. 2. Off-the-shelf OCR engine recognition accuracy may increase to acceptable rates if we provide pixel level text segmentation instead of raw pixels.

and lower-case variants of certain letters (e.g. "C" and "c") are tackled as a single class, and then differentiated using a typographic model that also serves to split the line into words. Finally, a combination of bi-gram and dictionary-based language model scores each text line hypothesis individually and the most probable hypothesis is selected. The authors further extended the text detection part of their method in several ways [10,11], increasing their end-to-end recognition results. In [12] they add a new inference layer to the recognition framework, where the best sequence selection is posed as an optimal path problem, solved by a standard dynamic programming algorithm, allowing the efficient processing of even more segmentation hypotheses.

Wang *et al.* [21] propose the use of Convolutional Neural Networks together with unsupervised feature learning to train a text detector and a character recognizer. The responses of the detector in a multi-scale sliding window, with Non-Maximal Suppression (NMS), give rise to text lines hypotheses. Word segmentation and recognition is then performed jointly for each text line using beam search. For every possible word the character recognizer is applied with an horizontal sliding window, giving a score matrix that (after NMS) can be used to compute an alignment score for all words in a small given lexicon. Words with low recognition score are pruned as being "non-text". Since the method is only able to recognize words in a small lexicon provided, in order to perform a more unconstrained recognition the authors make use of an off-the-shelf spell checking software to generate the lexicon given the raw recognition sequences.

In a very related work to the one presented in this paper Milyaev et al. [8] demonstrate that off-the-shelf OCR engines can still perform well on the scene text recognition task as long as appropriate image binarization is applied to input images. For this, they evaluate 12 existing binarization methods and propose a new one using graph cuts. Their binarization method is combined with an AdaBoost classifier trained with simple features for character/non-character classification. And the components accepted by the classifier are used to generate a graph by connecting pairs of regions that fulfill a set of heuristic rules on their distance and color similarity. Text lines obtained in such way are then split into words and passed to a commercial OCR engine[3] for recognition.

In [13] Neumann and Matas propose the detection of constant width strokes by convolving the gradient image with a set of bar filters at different orientations and scales. Assuming that characters consist in a limited number of such strokes a set of candidate bounding-boxes is created by the union of bounding boxes of 1 to 5 nearest strokes. Characters are detected and recognized by matching the stroke patterns with an Approximate Nearest Neighbor classifier trained with synthetic data. Each candidate bounding box is labelled with a set of character labels or rejected by the classifier. The non rejected regions are then agglomerated into text lines and the word recognition is posed as an optimal sequence search by maximizing an objective function that combines the probabilities of the character classifier, the probability of the inter-character spacing difference

[3] OCR Omnipage Professional, available at http://www.nuance.com/.

of each triplet, the probability of regions relative positioning, and the characters adjacency probability given by a 3-gram language model.

Yao et al. [22] propose an arbitrary oriented scene text detection and recognition method that extracts connected components in the Stroke Width Transform (SWT) domain [3]. Component analysis filters out non-text components using a Random Forest classifier trained with novel rotation invariant features. This component level classifier performs both text detection and recognition. Remaining character candidates are then grouped into text lines using the algorithm proposed by Yin *et al.* [23], and text lines are split into words using the method in [3]. Finally, the authors propose a modified dictionary search method, based on the Levenshtein edit distance but relaxing the cost of the edit operation between very similar classes, to correct errors in individual character recognition using a large-lexicon dictionary[4]. To cope with out-of-dictionary words and numbers, n-gram based correction [18] is used if the distance with closest dictionary word is under a certain threshold.

Bissacco *et al.* [1] propose a large-scale end-to-end method using the conventional multistage approach to text extraction. In order to achieve a high recall text detection the authors propose to combine the outputs of three different detection methods: a boosted cascade of Haar wavelets [19], a graph cuts based method similar to [15], and a novel approach based on anisotropic Gaussian filters. After splitting text regions into text lines a combination of two over-segmentation methods is applied, providing a set of possible segmentation points for each text line. Then beam search is used to maximize a score function among all possible segmentations in a given text line. The score function is the average per-character log-likelihood of the text line under the character classifier and the language model. The character classifier is a deep neural network trained on HOG features over a training set consisting of around 8 million examples. The output layer of the network is a softmax over 99 character classes and a noise (non-text) class. At test time this classifier evaluates all segmentation combinations selected by the beam search. The language model used in the score function to be optimized by the beam search is a compact character-level ngram model (8-gram). Once the beam search has found a solution the second level language model, a much larger distributed word-level ngram (4-gram), is used to rerank.

As a conclusion of the state of the art review we can see that the majority of reviewed methods make use of similar language models, based on a dictionary of frequent words and a character n-gram (usually a bi-gram). A much stronger language model has been proposed by Bissacco *et al.* [1]. On the other hand, while the system in [1] makes use of three different detection methods combined with an independent recognition module, in other cases both detection and recognition are treated together, e.g. using the same features for detection and recognition [22] or using multiple character detections as an over-segmentation cue for the recognition [10], giving rise to more compact and efficient methods.

[4] Word list is provided by the Microsoft Web N-Gram Service (http://webngram.research.microsoft.com/info/) with top 100k frequently searched words on the Bing search engine.

3 End-to-End Pipeline

In order to ensure that our results are reproducible we adopt a publicly available implementation of a well known algorithm for text detection. Concretely, the used OpenCV text module[5] implements, among others, the Class Specific Extremal Regions (CSER) algorithm initially proposed by Lukás Neumann & Jiri Matas [11], and the Exhaustive Search algorithm of the same authors [10].

The main idea behind Class-specific Extremal Regions is similar to the MSER in that suitable Extremal Regions (ERs) are selected from the whole component tree of the image. However, this technique differs from MSER in that selection of suitable ERs is done by a sequential classifier trained for character detection, i.e. dropping the stability requirement of MSERs and selecting class-specific (not necessarily stable) regions.

On the other hand, the Exhaustive Search algorithm was proposed in [10] for grouping ERs corresponding to character candidates into candidate text lines (for horizontally aligned text). The algorithm models a verification function that efficiently evaluates all possible ER sequences. The algorithm incorporates a feedback loop that allows to recover errors in the initial ER selection, by searching among the discarded ERs those that fit well with the detected text lines hypotheses. This feedback loop proves to be particularly important when doing the final recognition.

At this point it is fair to notice that the text detection implementation used in our experiments differs in several ways from the original work in [11]. Particularly, we work in a single channel projection (gray level image), i.e. we do not use different color channels, thus relying in a simplified pipeline that allows for faster computation at the expense of lower recall. Moreover, in our case the geometric normalization (if needed) and word splitting process is left to the OCR engine.

3.1 Text Recognition

The output of the described text detection algorithm is a set of text line hypotheses corresponding to groups of ERs. Those groups of ERs give rise to a pixel level segmentation of the detected text elements that can be directly fed to the OCR engine. Apart from that we further evaluate in our experiments the recognition performance by feeding three alternative inputs to the OCR: the gray scale cropped box of the detected lines, their Otsu's binarizations, and another binary image obtained with simple thresholding by setting as foreground all pixels with an intensity value in the range $(min(I_{ER_l}), max(I_{ER_l}))$, where I_{ER_l} is the set of intensity values of pixels corresponding to extremal regions of a given text line l.

We feed the results of the detection/segmentation pipeline to two well known off-the-shelf OCR engines: the open source project Tesseract[6] [17], and the commercial software ABBYY Fine Reader SDK[7]. The set-up for both OCR engines

[5] http://docs.opencv.org/trunk/modules/text/doc/erfilter.html.
[6] http://code.google.com/p/tesseract-ocr/.
[7] http://finereader.abbyy.com/.

is minimal: we set the recognition language to English and specify a closed list of characters (52 letters and 10 digits), we also set the OCR to interpret the input as a single text line, apart from that we use the default parameters.

The recognition output is filtered with a simple post-processing junk filter in order to eliminate garbage recognition, i.e. sequences of identical characters like "IIii" that may appear as a result of trying to recognize repetitive patterns in the scene. Concretely we discard the words in which more than half of their characters are recognized as one of "i", "l", or "I".

4 Experiments

We conduct all our experiments on the ICDAR2011 dataset in order to compare with all other methods that provide end-to-end results. The ICDAR2011 test set consists of 255 scene images with different sizes. The ground truth is defined at the word level using axis oriented bounding boxes and their corresponding text annotations. The evaluation protocol considers a valid recognition when the estimated word bounding box has at least 80 % recall with a ground-truth word and all its characters are recognized correctly (case sensitive).

Table 2 show the obtained results using the different end-to-end variants described before. Overall performance of *ABBYY* OCR engine is better than the obtained with *Tesseract*, and in both engines the better results are obtained with the CSER segmentation. *Tesseract* results with the Otsu thresholded image and the raw grey scale bounding boxes are exactly the same, which seems to indicate that the internal binarization method in *Tesseract* may be equivalent to the Otsu thresholding.

Table 2. End-to-end recognition evaluation in the ICDAR 2011 dataset comparing different segmentation approaches.

Method	Precision	Recall	F-score
CSER + ABBYY	59.2	39.3	47.2
CSER + Raw bbox + ABBYY	67.1	35.8	46.7
CSER + $(min(I_{ER_l}), max(I_{ER_l}))$ + ABBYY	65.9	36.1	46.6
CSER + Otsu + ABBYY	65.3	35.2	45.8
CSER + Tesseract	52.9	32.4	40.2
CSER + $(min(I_{ER_l}), max(I_{ER_l}))$ + Tesseract	60.4	29.3	39.5
CSER + Otsu + Tesseract	63.2	28.1	38.9
CSER + Raw bbox + Tesseract	63.2	28.1	38.9

Table 3 shows the comparison of the best obtained results of the two evaluated OCR engines with current state of the art. In both cases our results outperform [11] in total f-score while, as stated before, our detection pipeline is a simplified implementation of that method. However, it is important to notice

that our recall is in general lower that in the other methods, while it is our precision what makes the difference in f-score. A similar behaviour can be seen for the method of Milyaev *et al.* [8], which also uses a commercial OCR engine for the recognition. Such higher precision rates indicate that in general off-the-shelf OCR engines are doing very good in rejecting false detections.

Notice that Table 3 does not include the method in [1] because end-to-end results are not available. However, it would be expected to be in the top of the table if we take into account their cropped word recognition rates and their high recall detection strategy.

Table 3. End-to-end results in the ICDAR 2011 dataset comparing our proposed pipeline with other state-of-the-art methods. Methods marked with an asterisk evaluate on a slightly different dataset (ICDAR 2003).

Method	Precision	Recall	F-score
Milyaev *et al.*[8]	66.0	46.0	54.0
CSER + ABBYY	59.2	39.3	47.2
Yao *et al.* [22]	49.2	44.0	45.4
Neumann and Matas [13]	44.8	45.4	45.2
CSER + Tesseract	52.9	32.4	40.2
Neumann and Matas [12]	37.8	39.4	38.6
Neumann and Matas [11]	37.1	37.2	36.5
Wang *et al.* [21] *	54.0	30.0	38.0

It is indicative at this point comparing the recognition performance with the results of the detection module alone. For this we calculate precision and recall of line-level detection, i.e. we count a given detection bounding box as true positive if it overlaps more than 80 % with a ground truth bounding box, irrespective of the recognition output. The detection precision and recall obtained in this way are 56.7 and 73.5 % respectively. Notice that this detection rates are not comparable with the standard ICDAR evaluation framework for text localization, but just an indicative value in order to assess the effects of the recognition module in the final results.

Figures 3, 4, and 5 show qualitative results of the **CSER+Tesseract** pipeline. The end-to-end system recognizes words correctly in a variety of different situations, including difficult cases, e.g. where text appears blurred, or with non standard fonts. Common misspelling mistakes are, most of the time, due to missed diacritics, rare font types, and/or poor segmentation of some characters. Also in many cases the OCR performs poorly because the text extraction algorithm is not able to produce a good segmentation, e.g. in challenging situations or in cases where characters are broken in several strokes.

As part of our experiments we have also implemented a similar system to the **CSER+Tesseract** pipeline, but using MSER to be less computationally expensive, that reaches real-time end-to-end recognition with similar accuracy in a 640×480 video stream on a standard PC. For this experiment the whole

pipeline is set exactly as described in Sect. 3 but replacing the CSER character detector by the standard MSER algorithm. Table 4 show the speedup of this implementation compared to **CSER+Tesseract**, and the accuracies of both pipelines in the ICDAR2011 dataset. Obviously the required time to process a single image is not constant and depends, among other, on the number of detected text lines that are fed to the OCR engine. This makes difficult to measure the time performance in a real-time video stream, as an indicative result we provide the average processing time in a 640 × 480-resized version

Fig. 3. Qualitative results on well recognized text using the CSER+Tesseract pipeline.

Fig. 4. Common misspelling mistakes using the CSER+Tesseract pipeline are due to missed diacritics, rare font types, and/or poor segmentation of some characters.

Fig. 5. Common errors when segmentation is particularly challenging or characters are broken in several strokes.

of the ICDAR2011 dataset, which is 177 milliseconds. This processing time can be further reduced by using multi-threaded OCR workers, reaching an average frame-rate of $8 fps.$ in the ICDAR resized dataset with a 2.3 GHz quad-core i7 processor.

Table 4. End-to-end recognition performance in the ICDAR 2011 dataset comparing different region extraction approaches.

Method	Precision	Recall	F-score	Avg. time (ms.)
CSER + Tesseract	52.9	32.4	40.2	851.0
MSER + Tesseract	48.1	30.7	37.5	420.7

Finally, we have done a set of experiments in order to evaluate the specific impact of Tesseract's language model in the final recognition results of the **CSER+Tesseract** pipeline. The language model of the Tesseract engine has several components including an efficient segmentation search, a heuristic word rating algorithm, and a word bigram model. The main component is the set of different word dictionaries that are used for word rating in combination with different heuristic scores, like script/case/char-type consistency, and with the individual ratings of the character classifier. We have evaluated the **CSER+Tesseract** pipeline deactivating any dictionary-based correction. As can be seen in Table 5 the impact of the dictionary-based correction component accounts around 9 % of total f-score of the system.

Table 5. End-to-end recognition accuracy comparison in the ICDAR 2011 dataset by deactivating Tesseract's dictionaries.

Method	Precision	Recall	F-score
CSER + Tesseract	52.9	32.4	40.2
CSER + Tesseract (No-dict)	43.7	24.6	31.5

5 Discussion

We have evaluated two off-the-shelf OCR engines in combination with a scene text extraction method for the task of end-to-end text recognition in natural scenes, demonstrating that in such a pipeline conventional OCR solutions still perform competitively compared to other solutions specifically designed for scene text recognition. Moreover, our results are based in a simplified implementation of the text detection pipeline in [11], thus it is to be expected that a complete implementation of the original work would eventually improve the obtained results. Our findings, inline with other works reporting similar experiments [8], call to a discussion on the underlying factors of such results.

It is well known that complex language models are very useful in OCR applications where the character classifier has high error rates [18], for example in languages such as Arabic or Hindi. And this is certainly the case in scene text recognition, where state of the art methods are usually bounded to character recognition accuracies around 80 %.

Although document analysis from scanned documents in English language is not the same case, and thus traditional OCR for such task may rely much more in the character classifier confidence, it is still true that off-the-shelf engines, like the ones used in our experiments, have more developed language models than the usually found in scene text recognition methods.

As conclusion we can say that stronger language models, as found in off-the-shelf OCR engines or in the photoOCR system in [1], can make an important difference in the final recognition accuracy of end-to-end methods, and may be further investigated in the future.

On the other hand, regarding the usefulness of pixel level segmentation methods for scene text recognition we conclude that such techniques are nowadays able to provide state of the art accuracies for end-to-end recognition. It is true however that scene text is not always binarizable and that in such cases other techniques must be employed. But current segmentation methods combined with existing OCR technologies may produce optimal results in many cases, by taking advantage of more than 40 years of research and development in automated reading systems [4], e.g. all the accumulated knowledge of shape-based classifiers, and the state of the art in language modelling for OCR.

Acknowledgement. This project was supported by the Spanish project TIN2011-24631 the fellowship RYC-2009-05031, and the Catalan government scholarship 2013 FI1126. The authors want to thanks also Google Inc. for the support received through the GSoC project, as well as the OpenCV community, specially to Stefano Fabri and Vadim Pisarevsky, for their help in the implementation of the scene text detection module evaluated in this paper.

References

1. Bissacco, A., Cummins, M., Netzer, Y., Neven, H.: Photoocr: reading text in uncontrolled conditions. In: International Conference on Computer Vision (ICCV) (2013)
2. Chen, X., Yuille, A.L.: Detecting and reading text in natural scenes. In: Computer Vision and Pattern Recognition (CVPR) (2004)
3. Epshtein, B., Ofek, E., Wexler, Y.: Detecting text in natural scenes with stroke width transform. In: Computer Vision and Pattern Recognition (CVPR) (2010)
4. Fujisawa, H.: Forty years of research in character and document recognition an industrial perspective. Pattern Recogn. **41**, 2435–2446 (2008)
5. Gomez, L., Karatzas, D.: Multi-script text extraction from natural scenes. In: International Conference on Document Analysis and Recognition (ICDAR) (2013)
6. Karatzas, D., Shafait, F., Uchida, S., Iwamura, M., Gomez, L., Robles, S., Mas, J., Fernandez, D., Almazan, J., de las Heras, L.P.: ICDAR 2013 robust reading competition. In: International Conference on Document Analysis and Recognition (ICDAR) (2013)

7. Lucas, S.M., Panaretos, A., Sosa, L., Tang, A., Wong, S., Young, R.: ICDAR 2003 robust reading competitions. In: International Conference on Document Analysis and Recognition (ICDAR) (2003)
8. Milyaev, S., Barinova, O., Novikova, T., Kohli, P., Lempitsky, V.: Image binarization for end-to-end text understanding in natural images. In: International Conference on Document Analysis and Recognition (ICDAR) (2013)
9. Neumann, L., Matas, J.: A method for text localization and detection. In: Assian Conference on Computer Vision (ACCV) (2010)
10. Neumann, L., Matas, J.: Text localization in real-world images using efficiently pruned exhaustive search. In: International Conference on Document Analysis and Recognition (ICDAR) (2011)
11. Neumann, L., Matas, J.: Real-time scene text localization and recognition. In: Computer Vision and Pattern Recognition (CVPR) (2012)
12. Neumann, L., Matas, J.: On combining multiple segmentations in scene text recognition. In: International Conference on Document Analysis and Recognition (ICDAR) (2013)
13. Neumann, L., Matas, J.: Scene text localization and recognition with oriented stroke detection. In: International Conference on Computer Vision (ICCV) (2013)
14. Novikova, T., Barinova, O., Kohli, P., Lempitsky, V.: Large-lexicon attribute-consistent text recognition in natural images. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012. LNCS, vol. 7577, pp. 752–765. Springer, Heidelberg (2012)
15. Pan, Y.F., Hou, X., Liu, C.L.: Text localization in natural scene images based on conditional random field. In: International Conference on Document Analysis and Recognition (ICDAR) (2009)
16. Shahab, A., Shafait, F., Dengel, A.: ICDAR 2011 robust reading competition challenge 2: reading text in scene images. In: International Conference on Document Analysis and Recognition (ICDAR) (2011)
17. Smith, R.: An overview of the tesseract OCR engine. In: International Conference on Document Analysis and Recognition (ICDAR) (2007)
18. Smith, R.: Limits on the application of frequency-based language models to OCR. In: International Conference on Document Analysis and Recognition (ICDAR) (2011)
19. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: Computer Vision and Pattern Recognition (CVPR) (2001)
20. Wang, K., Babenko, B., Belongie, S.: End-to-end scene text recognition. In: ICCV (2011)
21. Wang, T., Wu, D.J., Coates, A., Ng, A.Y.: End-to-end text recognition with convolutional neural networks. In: International Conference on Pattern Recognition (ICPR) (2012)
22. Yao, C., Bai, X., Liu, W.: A unified framework for multi-oriented text detection and recognition. In: IEEE Transactions on Image Processing (TIP) (2014)
23. Yin, X.C., Yin, X., Huang, K., Hao, H.W.: Robust text detection in natural scene images. In: IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) (2013)