Scene Text Detection: A Survey

Yuqing Zhang 20911078 y3593zha@uwaterloo.ca

1. Introduction

In the field of computer vision, recent decades have witnessed a significant transformation, where deep learning techniques have proliferated over the past decade and replaced the traditional machine learning techniques. As an essential computer vision branch, text detection and recognition are primarily benefited by this deep learning fever.

There are mainly two critical concepts in the related field. One is Optical Character Recognition (OCR), used mostly to identify the text information on the scanned document. The other is Scene Text Recognition (STR), which refers to recognizing the text in the natural scenes. Obviously, STR is a more demanding task than OCR, as the text captured on the natural scenes has far richer forms than text on certificates or paper. Meanwhile, STR plays a substantial role in daily life, as it has been pervasively used in street view detection or visual search engines (Nayef et al., 2019).

There are mainly two sub-tasks in STR:

- text detection: detect or segment the area of the image which covered the text
- text recognition: recognize the text context (character) in the detected area

Each sub-task has been extensively investigated by many researchers. In this paper, our survey will be focused on scene text detection.

In the early stage, most techniques in scene text detection were inspired and adopted by the general object detection. With more profound development, more researchers have focused on different targets (like detecting irregular text), utilized the base features of the text, and made specific models. The main challenges stemmed from those aspects below:

- the text is varied in the fonts, colors, or size;
- the text line is often not horizontal. The curved or twisted text line is a barrier for researchers;
- it is hard to guarantee the quality of images. The text area may be too blur or deformed. Or sometimes the text background is too noisy that some specific texture near the text area interferes with the classification. Also, the background in different images can be extremely diverse.

In the following parts of this paper, the structure will be as follows: In section 2, I summarized the related algorithms based on their methodologies' taxonomy. Note that I did not follow a chronological order or performance order.In section 3, I did some analysis of the most state-of-the-art models and discussed some questions. Then finally, I talked about my own opinions about the promising future trend on this topic.

2. Survey

I referred to a previous literature review on STR (Long et al., 2020) to compare different techniques. I selected ten representative and well-performed models. They can be classified into two main classes.

2.1 Models Based On Object Detection

For general object detection, there were some powerful models, like SSD, YOLO, Fast R-CNN, and Faster R-CNN. However, their performance did not reach the expectations when the researchers directly put them in use on the scene text detection. The major reason is that scene text is often a fine-grained object with varying line lengths and aspect ratios.

Some early attempts to use deep convolution neural networks to detect text turned out to be substandard. The complex pipeline made the prediction process slower than expectations. Researchers started to turn their attention to general object detection since 2016, hoping to gain some penetrating insights and bring some vitality to text detection.

2.1.1 Models Inspired by One-Stage Object Detection

Inspired by one-stage object detection, in 2016, a fully-convolutional network, **TextBoxes** (Liao et al., 2016) has created. The architecture is mostly inherited by the famous VGG-16 network, where the last two fully-connected layers are replaced by convolutional layers by parameters downsampling. This idea was adopted from SSD, an emerging objection detection technique at that time. The final prediction output will be converted by Non-Maximum Suppression. Since text always has a large aspect ratio and text line is horizontal in most cases, the researchers used inception-style filters in the size 15 instead of the traditional 3*3 square filters.

The network used some default bounding boxes with fixed aspect ratios (1,2,3,5,7 and 10). A text-box layer will predict the default box's choice, the box size, and the box location. To solve the issue of dense horizontal distribution of the boxes and too sparse distribution in the vertical direction, the researchers add some vertical offsets. This ensures the boxes can match the text better.

However, it has some inevitable weaknesses.As TextBoxes adopts default bounding boxes with a fixed size and all the boxes are horizontal, the detection area is sometimes not perfect. The detection rectangle area may be larger than the actual text area and leave some blank space all around. Or it may cut the text area into unnecessary pieces. Moreover, it was incapable of detecting the text in an overexposed area or dark area. Also, TextBoxes is a word-based method. It may fall short when the character spacing is too large. The failure cases were selected in Figure 1(a).

Similar to TextBoxes, **EAST** (Zhou et al., 2017) also has a fully-convolutional architecture and uses Non-Maximum Suppression to generate the final prediction. But EAST only has the above two stages. As the full name, Efficient and Accuracy Scene Text detection implies, EAST has a simplified yet well-designed pipeline.

EAST adopts the idea of U-Net, a convolutional network designed mainly for medical image segmentation. It can keep a balance between utilizing the feature map and maintaining a high computation speed. Instead of using the large network VGG-16, East chose PVAnet as the base network, which is lighter in weight but has a much larger receptive field than VGG-16.

Apart from only rectangle regions can be detected, there are two advanced shapes that EAST can finally generate:

- rotated box region: the output will have text boxes and rotation angel two parts.

- Irregular quadrangle region: the outcome will be the coordinates of the quadrangle.

Besides the excellent flexibility in the geometric text regions, the designed model is also flexible in predicting in both word-level or text line level. It outperformed other methods in both prediction accuracy and running speed back to that time. The prediction result can be seen from Figure 1(b).There are still some limitations on EAST. If the text region is too large to exceed the instances limit, for example, long text lines traversing the images, EAST may still not detect them completely.

2.1.2 Models Inspired by Two-Stage Object Detection

Compared to one-stage object detection, two-stage object detection is more prevalent in being adapted by text detection methods. The first stage is to create a Region of Interest(ROI), and the second stage is to refine the results. The majority of them are region-proposal-based approaches.The last layer of RPN (Region Proposal Network) will extract features into two parallel layers (one for regression and the other for classification). RPN draws axis-aligned proposals and changes the scale and shape of the anchors to match the shape of the detecting object. RPN in general object detection often uses a rectangle to frame the object, regardless of the remaining part outside the object but inside the box. However, considering the next step of text recognition, the boundary between text and background should be more accurate, so that the noisy background will not inference and harm the recognition.

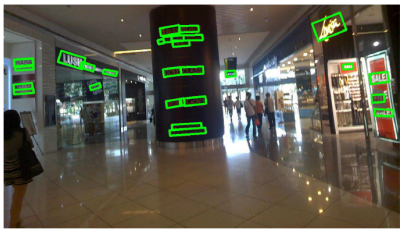
Apart from the methods discussed before that the models all do the detection in merely once perception, **LOMO** (LOok More Than Once) (Zhang et al., 2019) is dedicated to processing the localization for multiple times. There are three modules in LOMO:

- a direct regressor to generate a quadrangle in a per-pixel manner. This part is similar to EAST;
- a recursive refinement module to solve the problem of limited receptive fields. This process iteratively improves the result to be closer to the ground truth boxes;
- a shape expression module which is inspired by Mask R-CNN. The introduction of this module reconstructs the detection shape to detect the irregular text area as polygons.

LOMO can be regarded as an expert in detecting long text lines or curved text lines.

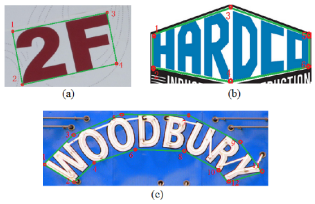
Similarly, the work Wang et al. (2019) also proposed for detecting irregular-shaped text. The authors proposed **adaptive text region representation** for the refinement part. The first part of the framework is still using the RPN in Faster R-CNN. Whereas for the second refinement stage, the previous works were based on CNN, while this work is using a recurrent neural network (RNN) to refine the proposals from the first region proposal network. For each time step in this process, a pair of boundary points are generated. This adaptive process makes the model powerful to output irregular bounding boxes. As this method does not need the pixel-level prediction, the speed is relatively fast (10 fps) compared to other methods supporting arbitrary shape texts.

(a) weakness of TextBoxes detectors.The green square is text area detected, while the red dotted box is the text area which the model failed to detect.



(b) results of EAST detectors. The detection shape can be quadrangle and the model tends to perform well even if the area is dark.

(c) results of LOMO. The red box is the final detection result.



(d) adaptive text region represented by 4 /6 /12 points

Figure 1: Comparison Of LOMO and adaptive text region representation

Figure 2: evolution of methods inspired by general object detection

From the Figure 2 we can witness the evolution of text detection methods inspired by object detection, from axis-aligned rectangles (TextBoxes) to rotated rectangles or quadrangles (EAST), and irregular polygons (LOMO and adaptive text region representation).

In summary, compared with the earliest attempts in deep learning, methods inspired by object detection enjoy a simpler pipeline, and they can perform far better. However, one-stage detectors fall short when dealing with the irregular text line, while the running speed of two-stage detectors is understandably slower. I summarized some comparison of the methods' properties above in the Table 1.

Model Name	Related Object Detection Model	backbone network	Area Shape	Long Text	Curved Text	Detection Level
TextBoxes	SSD	VGG16	rectangle	Bad	Bad	word
East	U-Net	PVAnet	rotated rectangle or quadrangle	Not so good	Not so good	word / text line
LOMO	Mark R-CNN	Resnet50 FPN	polygons	Good	Good	region
adaptive text region	faster-rcnn	VGG16	irregular	Good	Good	region

Table 1: Comparison between researchers inspired by object detection

2.2 Models Based on Sub-Text Components

Since text is homogenous as a whole. Any segmenting part of the text is still considered as text. This property encouraged researchers to design more algorithms that predict only sub-text components. After incorporating the prediction results, the models can reconstruct the whole text instances and generate a more accurate result. These models thus have more adaptability in solving long-text or curved-text challenges. The methods can be classified into different levels, such as Pixel-level, Component-level, and Character-level.

2.2.1 Pixel-Level

The core idea and challenges in Pixel-Level is to divides each pixel into three categories: text, border, and background. Predicting the border is the most challenging part. It is hard for the model to determine whether two adjacent pixels belong to the same text instance via the previous semantic segmentation. This is largely solved in the work **PixelLink** (Deng et al., 2018), which can extract the final text bounding boxes by Instance Segmentation instead of location regression. The previous methods based on objection detection always suffer from the limitation of receptive fields of CNN. In contrast, PixelLink has a much lesser requirement on receptive fields as it does not need to observe too many statuses from the feature map.

2.2.2 Component-Level

As a representative of works in Component-level, Connectionist Text Proposal Network (**CTPN**) (Tian et al., 2016) consider the text as a sequence of multi-level components. Not only characters, strokes, words, text lines, and text regions can all be regarded as components. Utilizing the relationship in the sequence, the authors proposed to stack an LSTM structure on the top of CNN. Aiming at the horizontal text detection, CTPN uses a group of anchors with equal width and different heights to locate the text position as shown in Figure 3(a). By the filters in the convolutional feature maps, an array of fine-scale text proposals are produced for each text line.

However, the use of anchors is a double-edged sword. Along with the addition of LSTM, CTPN enjoys outstanding performance on horizontal text detection. But the detection effect of irregular text is not as good as expected due to the limitation of the framework design with anchors.

Inherited by the idea of fine-scale text proposals in CTPN, Segment Linking (**SegLink**) (Shi et al., 2017) solved the problem in CTPN of the inability to detect non-horizontal text. SegLink uses a CNN network and estimates the offsets of default boxes to extract segments(oriented boxes). The most core process of the algorithm is to establish links within segments. To define which segments need to be combined, the depth-first search is used in the final stage of this model. This novel technique is superior in detecting horizontal, oriented, and multi-lingual text. A minor weakness is that SegLink always fails in detecting text where the character space is too large because it is hard to build the links.

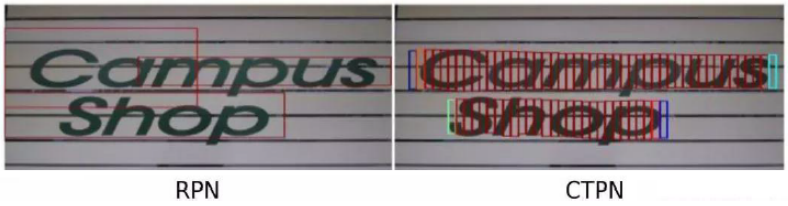
Similar to SegLink, one very recent research (Zhang et al., 2020) proposed a novel model that also builds links and bridges between the text proposals. What makes this research extraordinary is that this network is using **Graph Convolutional Network** to deal with the deep relational reasoning in those text proposals according to their geometry attributes. It is end-to-end trainable by building a local graph between the text proposals network and the deep relational network. This GCN method is an expert in detecting text of arbitrary shape, especially curved text. The excellent result can be seen in Fig. 3(c).

An interesting method is called **TextSnake**. As the name has shown, it is a flexible representation specially designed for detecting curved text and irregular text. It uses some sliding and overlapping circles whose centers are on the text center line (Fig.3(b)). Then the text center line is masked by the text region map to reconstruct the text instance. A fully convolutional network constitutes the network architecture. Three steps of predicting the text center line are centralizing, striding, and sliding. Those overlapping circles are in different radius and geometry attributes, thus making the predicted text region fit the ground-truth text regions very well.

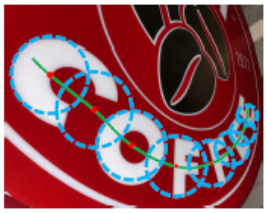
2.2.3 Character-Level

Techniques in the Character-Level are new emerging these years, yet they are powerful and effective. Character Region Awareness For Text detection (**CRAFT**) (Baek et al., 2019) proposed a new method to predict the location of each character and incorporate them into a whole text line using the affinity between characters. The network produced Gaussian Heatmaps for each character prediction (in fig.3(d)). Additionally, as most of the dataset did not provide the information of character locations, the researchers spent much effort in proposing another framework to estimate the character annotations. This work can outperform almost all the other methods on some benchmark datasets.

In summary, detection based on sub-text components in various levels can fit the ground-truth text instance better. Some methods are excellent in dealing with curved text detection. They all have similar architectures: first, there is a CNN to predict the location of sub-text components in different levels; then in the second step, various frameworks are proposed to classify pixels or segments, or characters into text instances. A minor deficiency is that the post-processing step of most models may be prone to noise, thus leading to some overfitting.



(a) the idea of CTPN. Left is the result of traditional RPN and right is the result of CTPN. It uses equal-width anchors to produce text proposals.



(b) the idea of TextSnake. It uses some overlapping disks whose centers are on the text center line.

(c) the result of GCN

(d) the idea of CRAFT

Figure 3: Comparison of works based on sub-text components

3. Analysis

The methods I selected above were all proposed after 2016 and the most top-level techniques in the field of text detection. Several of them are extraordinary. Some benchmark datasets are having different features are frequently used. For ICDAR Robust Reading Competition(2013 and 2017), CRAFT(Character Region Awareness For Text detection) has the best performance when referring to the F1 score (95.2 and 73.9, respectively). But the result of CRAFT in ICDAR 2015 (with a F1 = 89.7) was outperformed by adaptive text region representation proposed in Wang et al. (2019) (with a F1 = 89.7). Graph Convolutional Network method performs the best (has the best F1 score of 85.43) when dealing with irregular text, as shown in the results in other famous datasets such as Total Text (a dataset contained a large proportion of curved text). Consider the overall results of synthesis, I assume CRAFT and the Graph Convolutional Network method are the state-of-the-art methods. The detailed result table can be seen in the table 2.

Note that, all the datasets and the corresponding evaluation protocols are not perfect. According to Baek et al. (2019), it is widely acknowledged that inevitable discrepancies may occur when the researchers did experiments on the same dataset but use different subsets. In addition, the majority of datasets use IoU for detection evaluation. If the detected bounding box has more than 50% overlap (intersection over union, IoU) with the ground truth box, the detection is regarded as positive and accurate.

However, the IoU method may be problematic. The score for each image is assigned to 1 as long as the IoU proportion is larger than a designated threshold (mostly 0.5). Figure is a case that both methods can have a score of 1 for this picture, but the method on the right is intuitively better than the left. As many methods are performing closely well in

text detection, this protocol may be adverse for those which can fit the ground-truth boxes perfectly.

	ICDAR 13			ICDAR 15			ICDAR 17 MLT			Total-Text			FPS
Model Name	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	
TextBoxes	0.88	0.83	0.85										2.3
East				0.8327	0.7833	0.8072				0.5	0.362	0.42	
LOMO				0.878	0.876	0.877	0.802	0.672	0.731				
Adaptive text region	0.937	0.897	0.917	0.892	0.86	**0.876**				0.809	0.762	0.785	
PixelLink				0.855	0.82	0.837							3
CTPN	0.93	0.83	0.88	0.74	0.52	0.61							
SegLink	0.877	0.83	0.853	0.747	0.765	0.756				0.303	0.238	0.267	**20.6**
Graph Network				0.8853	0.8469	0.8656	0.7409	0.6104	0.6731	0.8654	0.8493	**0.8573**	
TextSnake				0.849	0.804	0.826				0.827	0.745	0.784	1.1
CRAFT	0.974	0.931	**0.952**	0.898	0.848	0.869	0.806	0.682	**0.739**	0.876	0.799	0.836	8.6

Table 2: Results of methods discussed in this report on different benchmark datasets, where P is the Precision, R is the Recall and F1 score equals 2*

Since scene text detection and recognition are complementary sub-tasks. In word-level detection, a large proportion of wrong or missed detection is that the detectors fail to detect the space between two words or other text components. Using the result in the recognition part may make up for the deficiency in the detection part. For example, when recognizing two words in one detected area, the recognition model may convey this feedback to the detection model and cut the detection part into two pieces. Merely focusing on the detection part may be too risky.

4. Conclusion

This report has summarized some representative techniques for scene text detection. After entering the deep learning era, this topic has much promising progress. At first, most techniques adapted the framework of general object detection and fell short when encountering the oriented text, long text, or curved text. But with the development in recent years, some methods were designed for solving these challenges.

For future research, I list several recommendations:

- In my survey, I found many works did not provide an end-to-end system for text detection. From my point of view, I suppose more researches can spend effort on developing an integrated system in text detection and recognition.
- Generalization and Robustness are two crucial concepts but seemed not to be valued in the previous researches. As in some real-world scenarios, it is hard to predict what kind of data the user will send to the application. So maintaining a good level of adaptability to diverse environments. I recommend more using one benchmark dataset to train and another dataset to evaluate the performance in the future.
- For the benchmark datasets, many improvements are still worthy. Some previous datasets did not provide the information of punctuations, which can be problematic. More datasets can be annotated at the character level, providing the ground-truth character and the bounding boxes location. There is more need for multilingual datasets with more languages as well.

References

Youngmin Baek, Bado Lee, Dongyoon Han, Sangdoo Yun, and Hwalsuk Lee. Character region awareness for text detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9365–9374, 2019.

Dan Deng, Haifeng Liu, Xuelong Li, and Deng Cai. Pixellink: Detecting scene text via instance segmentation. *arXiv preprint arXiv:1801.01315*, 2018.

Minghui Liao, Baoguang Shi, Xiang Bai, Xinggang Wang, and Wenyu Liu. Textboxes: A fast text detector with a single deep neural network. *arXiv preprint arXiv:1611.06779*, 2016.

Shangbang Long, Xin He, and Cong Yao. Scene text detection and recognition: The deep learning era. *International Journal of Computer Vision*, pages 1–24, 2020.

Nibal Nayef, Yash Patel, Michal Busta, Pinaki Nath Chowdhury, Dimosthenis Karatzas, Wafa Khlif, Jiri Matas, Umapada Pal, Jean-Christophe Burie, Cheng-lin Liu, et al. Icdar2019 robust reading challenge on multi-lingual scene text detection and recognition—rrc-mlt-2019. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1582–1587. IEEE, 2019.

Baoguang Shi, Xiang Bai, and Serge Belongie. Detecting oriented text in natural images by linking segments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2550–2558, 2017.

Zhi Tian, Weilin Huang, Tong He, Pan He, and Yu Qiao. Detecting text in natural image with connectionist text proposal network. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 56–72, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46484-8.

Xiaobing Wang, Yingying Jiang, Zhenbo Luo, Cheng-Lin Liu, Hyunsoo Choi, and Sungjin Kim. Arbitrary shape scene text detection with adaptive text region representation, 2019.

Chengquan Zhang, Borong Liang, Zuming Huang, Mengyi En, Junyu Han, Errui Ding, and Xinghao Ding. Look more than once: An accurate detector for text of arbitrary shapes, 2019.

Shi-Xue Zhang, Xiaobin Zhu, Jie-Bo Hou, Chang Liu, Chun Yang, Hongfa Wang, and Xu-Cheng Yin. Deep relational reasoning graph network for arbitrary shape text detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9699–9708, 2020.

Xinyu Zhou, Cong Yao, He Wen, Yuzhi Wang, Shuchang Zhou, Weiran He, and Jiajun Liang. East: An efficient and accurate scene text detector. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.