# Shape Robust Text Detection with Progressive Scale Expansion Network

Wenhai Wang, Enze Xie, Xiang Li, Wenbo Hou, Tong Lu*, Gang Yu, Shuai Shao

CVPR LONG BEACH CALIFORNIA June 16-20, 2019


## Abstract:

- There exists two challenges which prevent the algorithm into industry applications. On the one hand, most of the state-of-art algorithms require quadrangle bounding box which is in-accurate to locate the texts with arbitrary shape. On the other hand, two text instances which are close to each other may lead to a false detection which covers both instances. To address these two challenges, in this paper, we propose a novel Progressive Scale Expansion Network (PSENet), which can precisely detect text instances with arbitrary shapes.


## Motivation:

- For the regression-based approaches, the text targets are in the forms of quadrangles and fail to deal with the text instance with arbitrary shapes (see Fig. 1 (a)).
- For the segmentation-based approaches, they can locate the text instance based on pixel-level classification, but they are difficult to separate the text instances which lying closely (see Fig. 1 (b)).

## Contribution:

- we propose an arbitrary-shaped text detection framework. The key points of the framework are "kernel" and "rebuilding text instance from kernel".
- we propose an algorithm to rebuild the text instance, namely, progressive scale expansion (PSE) algorithm, which can fast reconstruct the text instance from kernel.

(a) (b)

Fig. 1 The results of different methods

## Method (see Fig. 2):

1. We use ResNet-50 as the backbone of PSENet and concatenate low-level texture feature with high-level semantic feature (see Fig. 2 feature map $F$);
2. The feature map $F$ is projected into n branches to produce multiple segmentation results $S_1, S_2, \ldots, S_n$, Each $S_i$ is one segmentation mask for all the text instances at a certain scale;
3. We use progressive scale expansion algorithm (see Fig.3) to gradually expand all the instances' kernels in $S_1$, to their complete shapes in $S_n$, and obtain the final detection results as $R$.

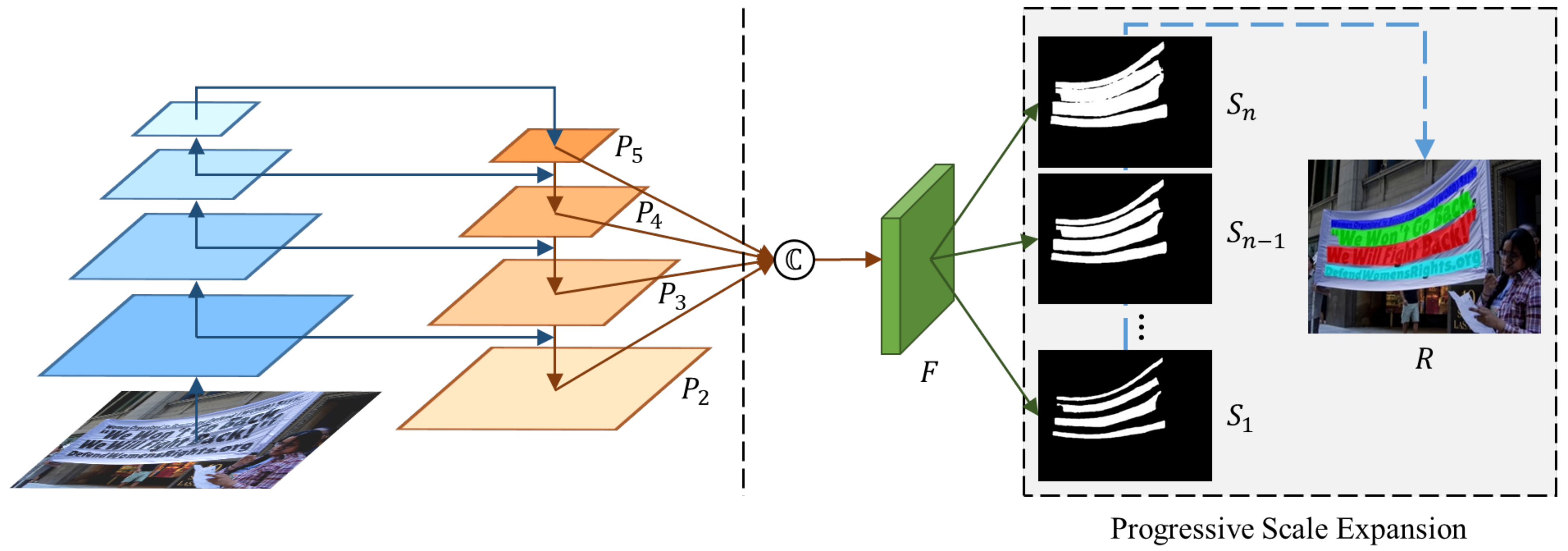


Fig. 2 The overall pipeline.

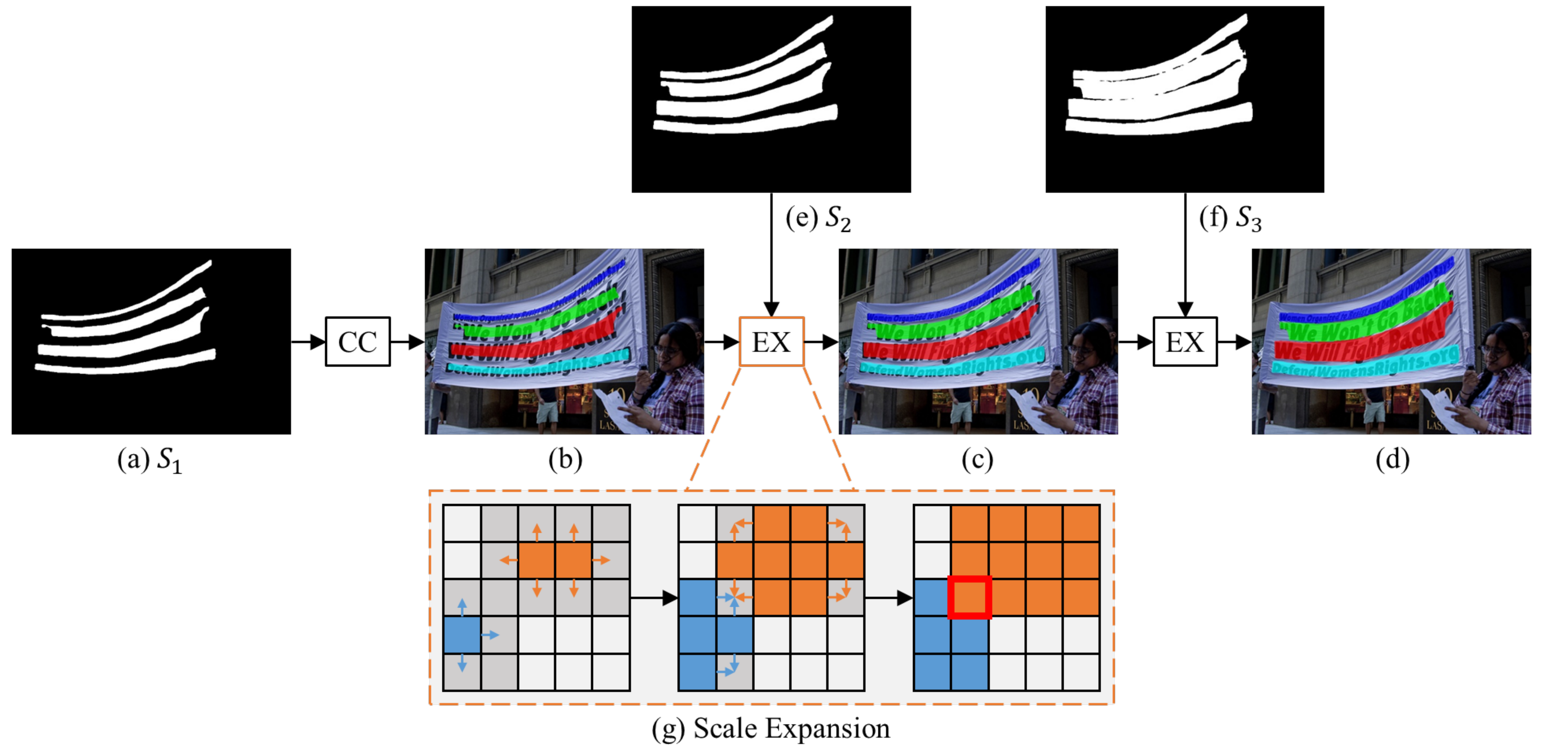


Fig. 3 The procedure of PSE. "CC" refers to the function of finding connected components. "EX" represents the scale expansion algorithm.

## Label Generation (see Fig. 4):

- If we consider the scale ratio as $r_i$, the margin $d_i$ between $p_n$ and $p_i$ can be calculated as:

$$d_i = \frac{\mathrm{Area}(p_n) \times (1 - r_i^2)}{\mathrm{Perimeter}(p_n)}$$

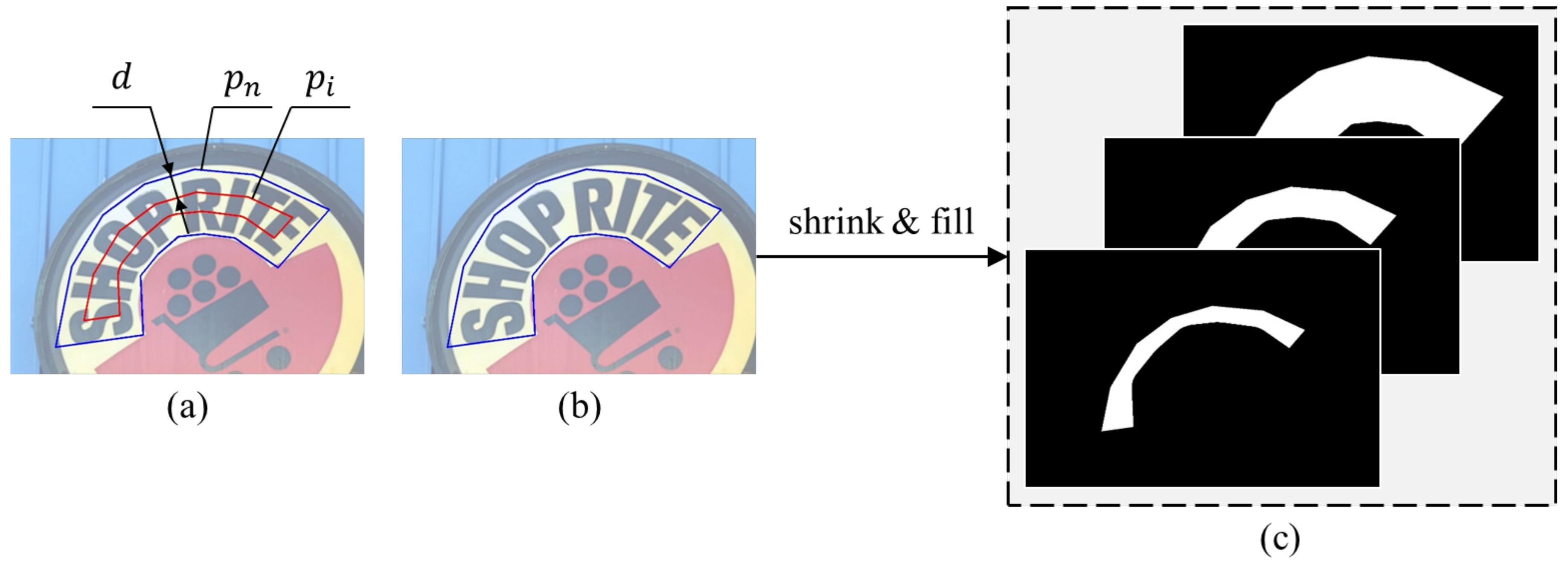


Fig. 4 The illustration of label generation.

## Results:

| Method | Ext | CTW1500 | | | |
|---|---|---|---|---|---|
| | | P | R | F | FPS |
| CTPN [36] | - | 60.4* | 53.8* | 56.9* | 7.14 |
| SegLink [32] | - | 42.3* | 40.0* | 40.8* | 10.7 |
| EAST [41] | - | 78.7* | 49.1* | 60.4* | **21.2** |
| CTD+TLOC [24] | - | 77.4 | 69.8 | 73.4 | 13.3 |
| TextSnake [26] | ✓ | 67.9 | 85.3 | 75.6 | - |
| PSENet-1s | - | 80.57 | 75.55 | 78.0 | 3.9 |
| PSENet-1s | ✓ | 84.84 | 79.73 | **82.2** | 3.9 |
| PSENet-4s | ✓ | 82.09 | 77.84 | 79.9 | 8.4 |

| Method | Ext | Total-Text | | | |
|---|---|---|---|---|---|
| | | P | R | F | FPS |
| SegLink [32] | - | 30.3 | 23.8 | 26.7 | - |
| EAST [41] | - | 50.0 | 36.2 | 42.0 | - |
| DeconvNet [2] | - | 33.0 | 40.0 | 36.0 | - |
| TextSnake [26] | ✓ | 82.7 | 74.5 | 78.4 | - |
| PSENet-1s | - | 81.77 | 75.11 | 78.3 | 3.9 |
| PSENet-1s | ✓ | 84.02 | 77.96 | **80.87** | 3.9 |
| PSENet-4s | ✓ | 84.54 | 75.23 | 79.61 | **8.4** |

Table 1 The results on CTW1500 and Total-Text.

| Method | Res | F | Time consumption | | | FPS |
|---|---|---|---|---|---|---|
| | | | backbone(ms) | head(ms) | PSE(ms) | |
| PSENet-1s (ResNet50) | 1280 | 82.2 | 50 | 68 | 145 | 3.9 |
| PSENet-4s (ResNet50) | 1280 | 79.9 | 50 | 60 | 10 | 8.4 |
| PSENet-4s (ResNet50) | 960 | 78.33 | 33 | 35 | 9 | 13 |
| PSENet-4s (ResNet50) | 640 | 75.6 | 18 | 20 | 8 | 21.65 |
| PSENet-4s$^{\dagger}$ (ResNet18) | 960 | 74.30 | 10 | 17 | 10 | 26.75 |

Table 2 Time consumption of PSENet on CTW-1500.

**Code: https://github.com/whai362/PSENet**