Text Perceptron: Towards End-to-End Arbitrary-Shaped Text Spotting

ID:893

Supplements

Derivation of TPS

The derivation of TPS refers to (Shi et al. 2016). We define the TPS transformation that receives input images or feature maps U with size of $U \in R^{H\times W\times C}$, where H, W, C denote the region's height, width and channels separately, and it generates the output as $V \in R^{H'\times W'\times C}$. TPS learns $2\times N$ fiducial points in the input feature map, i.e., $P \in R^{N\times 2}$. We denote the fiducial points in the output feature map as $P' \in R^{N\times 2}$ (corresponding to P^* in the main paper), which are preset constant value.

Forward Process For every mapped fiducial points in output feature map P', denoted as $p'_i = (x'_i, y'_i)$, we calculate the TPS transform like:

$$\begin{pmatrix} x_i \\ y_i \end{pmatrix} = TPS(p'_i) = \begin{pmatrix} \Delta_{p'}^{-1} & \begin{bmatrix} P^T \\ 0^{3\times 2} \end{bmatrix} \end{pmatrix}^T \cdot \left[1, x'_i, y'_i, d^2_{i,1}, \ln d^2_{i,1}, d^2_{i,n}, \ln d^2_{i,n}, ..., d^2_{i,N}, \ln d^2_{i,N}\right]^T \tag{1}$$

where $\Delta_{p'}^{-1}$ is a matrix with size of $(N+3)\times(N+3)$, and computed by the transformed fiducial points in P^*:

$$\Delta_{p'} = \begin{bmatrix} K^{N\times N} & P^{3\times N} \\ P^{3\times N T} & 0^{3\times 3} \end{bmatrix} \tag{2}$$

where

$$P^{3\times N} = \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ ... & ... & ... \\ 1 & x_k & y_k \end{bmatrix} \tag{3}$$

$$K^{N\times N} = \begin{bmatrix} 0 & U(r_{12}) & ... & U(r_{1N}) \\ U(r_{21}) & 0 & ... & U(r_{2N}) \\ ... & ... & ... & ... \\ U(r_{N1}) & U(r_{N2}) & ... & 0 \end{bmatrix}, \quad U(r_{ik}) = d^2_{i,k} \ln d^2_{i,k} \tag{4}$$

Note that all computations are conducted after the normalization between $[-1, 1]$.

Therefore, for every sampled pixel in the input image, we can calculate a mapped target pixel. Then, the output image can be formed as:

$$V_i^c = \sum_h^H \sum_w^W U_{hw}^c k(x_i - w; \Phi_x) k(y_i - h; \Phi_y), \quad \forall i \in [1...H'W'], \forall c \in [1, ...C] \tag{5}$$

where Φ_x and Φ_y are the parameters of sampling kernel function $k(.)$. For example, if we adopt the bilinear interpolation algorithm, it can be calculated as:

$$V_i^c = \sum_h^H \sum_w^W U_{hw}^c max(0, 1 - |x_i - w|) k(0, 1 - |y_i - h|), \quad \forall i \in [1...H'W'], \forall c \in [1, ...C] \tag{6}$$

Backward Process We should define the gradients between the output loss and input images U and their sampled points. The computation of sampling is differentiable. For bilinear interpolation, the partial differential can be formed as:

$$\frac{\partial V_i^c}{\partial U_{hw}^c} = \sum_h^H \sum_w^W max(0, 1 - |x_i - w|) max(0, 1 - |y_i - h|), \quad \forall i \in [1...H'W'], \forall c \in [1, ...C] \tag{7}$$

$$\frac{\partial V_i^c}{\partial x_i^c} = \sum_h^H \sum_w^W U_{hw}^c max(0, 1 - |y_i - h|) \begin{cases} 0, |x_i - w| \geq 1 \\ 1, w \geq x_i \\ -1, w < x_i \end{cases}, \quad \forall i \in [1...H'W'], \forall c \in [1, ...C] \tag{8}$$

$$\frac{\partial V_i^c}{\partial y_i^c} = \sum_h^H \sum_w^W U_{hw}^c max(0, 1 - |x_i - w|) \begin{cases} 0, |y_i - h| \geq 1 \\ 1, h \geq y_i \\ -1, h < y_i \end{cases}, \quad \forall i \in [1...H'W'], \forall c \in [1, ...C] \tag{9}$$

Above formulas calculate the loss gradients to input feature maps as well as the sampled points, which is corresponding

to the $\frac{\partial \mathcal{R}^*}{\partial \mathcal{R}}$ in formula-(10) of the main paper. Furthermore, we also need to get the gradients to the fiducial points, i.e., $\frac{\partial \mathcal{R}^*}{\partial \mathcal{R}} \frac{\partial \mathcal{R}}{\partial P}$ in formula-(11) of the main paper. For TPS, it is computed as the multiplication of two matrix, and so is differentiable.

$$\partial \begin{pmatrix} \Delta_{p'}^{-1} & \begin{bmatrix} P^T \\ 0^{3\times 2} \end{bmatrix} \end{pmatrix}^T = \partial \begin{pmatrix} x_i \\ y_i \end{pmatrix} \cdot \left[1, x_i', y_i', d_{i,1}^2, \ln d_{i,1}^2, d_{i,n}^2, \ln d_{i,n}^2, ..., d_{i,N}^2, \ln d_{i,N}^2\right]^T \tag{10}$$

$$\partial \begin{pmatrix} \begin{bmatrix} P^T \\ 0^{3\times 2} \end{bmatrix} \end{pmatrix}^T = \partial \begin{pmatrix} x_i \\ y_i \end{pmatrix} \cdot \Delta_{p'}^{-1} \cdot \left[1, x_i', y_i', d_{i,1}^2, \ln d_{i,1}^2, d_{i,n}^2, \ln d_{i,n}^2, ..., d_{i,N}^2, \ln d_{i,N}^2\right]^T \tag{11}$$

In this way, we directly obtain the $\frac{\partial x_i}{\partial P}$ and $\frac{\partial y_i}{\partial P}$ from above computation. According to the chain rule, the gradients to those fiducial points are formed as:

$$\frac{\partial V_i^c}{\partial P} = \frac{\partial V_i^c}{\partial x_i^c} \cdot \frac{\partial x_i^c}{\partial P} \quad \text{or} \quad \frac{\partial V_i^c}{\partial P} = \frac{\partial V_i^c}{\partial y_i^c} \cdot \frac{\partial y_i^c}{\partial P}, \quad \forall i \in [1...H'W'], \forall c \in [1, ...C] \tag{12}$$

Note that the fiducial points in original STN (Shi et al. 2016) are only supervised by the recognition, which, in practice are hard to be optimized. In our settings, the fiducial points of input images are generated by the help of the proposed detector, which can be optimized much more efficient and effective.

Detection Results Comparison on IC15

Method	Precision	Recall	F-Measure	FPS
Zhang et al.(2016)	71.0	43.0	54.0	0.5
He et al. (2017b)	82.0	80.0	81.0	1.1
SSTD (2017a)	80.0	73.0	77.0	7.7
EAST (2017)	83.6	73.5	78.2	**13.2**
He et al. (2018)	87.0	86.0	87.0	-
FOTS (2018)	91.0	85.2	**88.0**	7.8
TextSnake* (2018)	84.9	80.4	82.6	1.1
TextField* (2019)	84.3	80.5	82.4	5.2
FTSN* (2018)	88.6	80.0	84.1	-
SPCNet* (2019)	88.7	85.8	87.2	-
PSENet-1s* (2019a)	86.9	84.5	85.7	1.6
LOMO* (2019)	83.5	**91.3**	87.2	-
Wang et al.* (2019b)	86.0	89.2	87.6	10.0
Tian et al.* (2019)	85.0	88.3	86.6	-
CRAFT* (2019)	84.3	89.8	86.9	8.6
TextNet* (2018)	89.4	85.4	87.4	-
Mask TextSpotter* (2018)	91.6	81.0	86.0	4.8
Ours (2-stage)	91.6	81.8	86.4	8.8
Ours (End-to-end)	**92.3**	82.5	87.1	8.8

Table 1: Detection Results on IC15. Superscript '*' means that the method considered the detection of irregular text.

Table 1 shows the performance of the detection task on IC15. Our method obtains the highest precision and a relative fast running speed. Although it doesn't reach the highest F-measure among all of the methods, the result still demonstrates that our method is compatible with the of the previous state-of-the-art methods.

Visualization Result

Figure 1 and Figure 2 demonstrate some visualization results from IC13/IC15/Total-Text/CTW1500 dataset. Text Perceptron shows its powerful ability in catching the reading order of scene text, and with the help of fiducial points which can further recognize text in a much simpler way. From the segmentation results, we can also find many of text-like false positives have been filtered out due to the missing of *head* or *tail* boundary. This means the features of *head* or *tail* boundaries may have different semantic information with that of the *center* region.

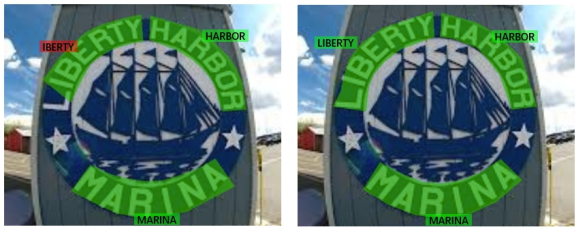

Figure 3: Example of comparison between two-stage evaluation and end-to-end training evaluation.

Figure 3 demonstrates an example that how end-to-end training boosts the localization of fiducial points. Only supervised by detection annotation sometimes cannot obtain a group of perfect positions of fiducial points, although it meets the requirement of detection. In end-to-end training manner, the positions of fiducial points may be dynamically tuned by the supervision of following recognition task, which finally generates a reasonable recognition result.

Failure Samples

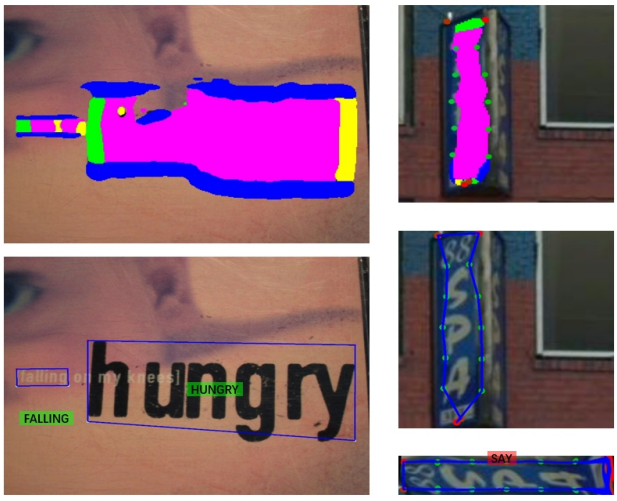

Figure 4: Visualization of some failure samples.

We illustrate some failure samples that are difficult for Text Perceptron, as shown in Figure 4.

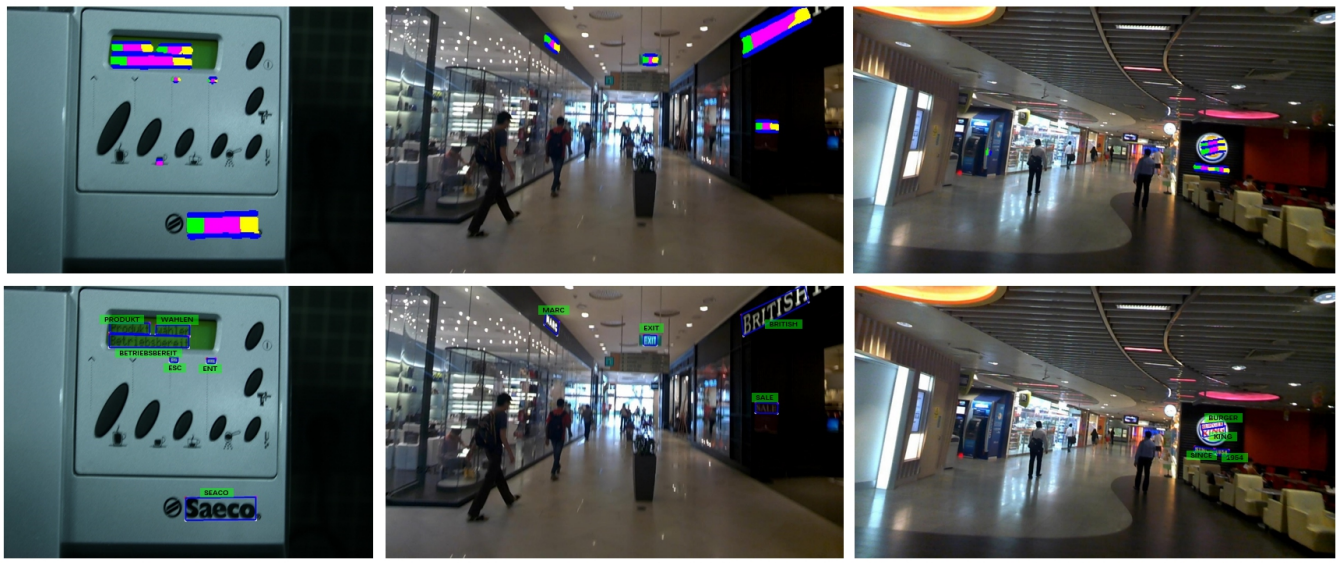

Figure 1: Visualization result on IC13 and IC15. The first row displays the segmented results and the second row shows the end-to-end results.

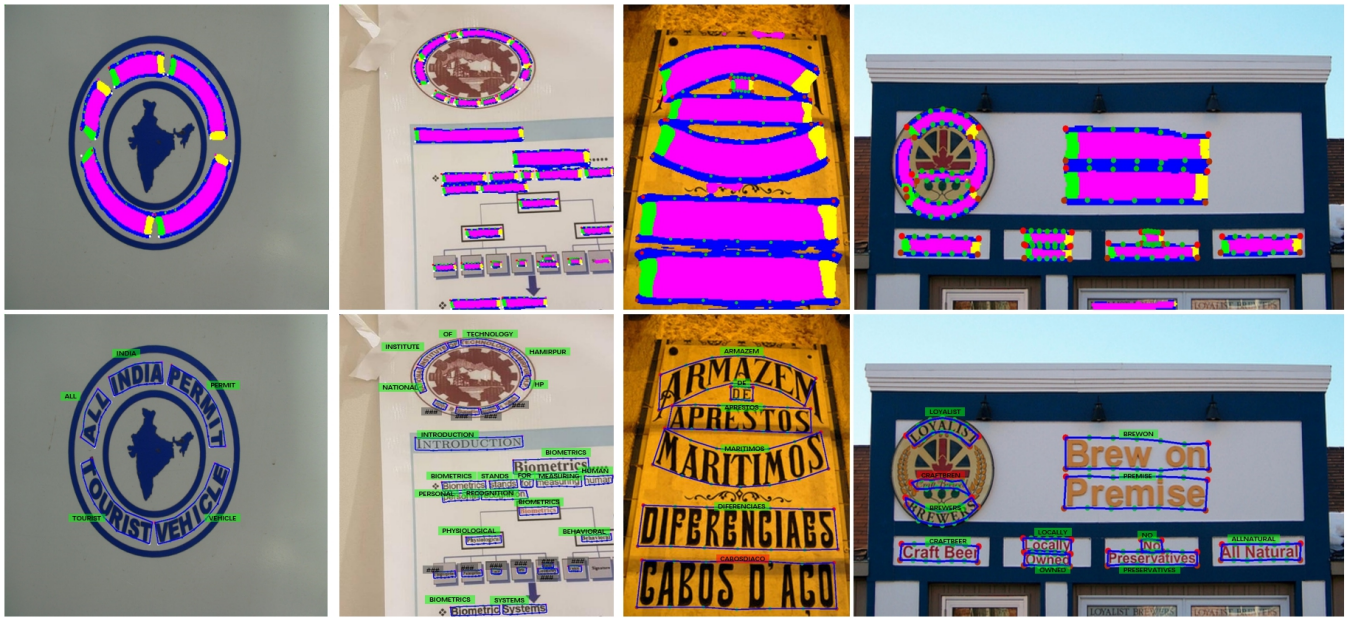

Figure 2: Visualization result on Total-Text and CTW1500. The first row displays the segmented results and the second row shows the end-to-end results. Fiducial points are also visualized as colored points on text boundaries.

Overlap text. It is a common tough task for segmentation-based detection methods. Pixels belong to the *center* text region for one text instance may also become the boundary region for another one. Even though our orderly overlaying strategy allows pixels to have multiple classes and makes boundary pixels have higher priority than *center* text pixels, which encourages inner instance to be separated from the outer instance. But experiments found that many times, the boundaries of inner instance cannot be fully recalled to embrace such instance, and connecting between *center* pixels will result in the failure of detecting such inner an instance.

Recognition of vertical instance. On the one hand, vertical texts appear in little frequency in the common datasets. One the other hand, although Text Perceptron can read vertical instances from left to right, it is still a challenge for recognition algorithm to distinguish whether the instance is a horizontal text or a 'lying-down' vertical one. Therefore, there are some correctly detected instances cannot be recognized right. It is also a common difficult problem for all existing recognition algorithms.

References

Baek, Y.; Lee, B.; Han, D.; Yun, S.; and Lee, H. 2019. Character Region Awareness for Text Detection. In *CVPR*.

Dai, Y.; Huang, Z.; Gao, Y.; Xu, Y.; Chen, K.; Guo, J.; and Qiu, W. 2018. Fused Text Segmentation Networks for Multi-oriented Scene Text Detection. In *ICPR*, 3604–3609.

He, P.; Huang, W.; He, T.; Zhu, Q.; Qiao, Y.; and Li, X. 2017a. Single Shot Text Detector with Regional Attention. In *ICCV*, 3047–3055.

He, W.; Zhang, X.-Y.; Yin, F.; and Liu, C.-L. 2017b. Deep Direct Regression for Multi-Oriented Scene Text Detection. In *ICCV*, 745–753.

He, T.; Tian, Z.; Huang, W.; Shen, C.; Qiao, Y.; and Sun, C. 2018. An End-to-End TextSpotter with Explicit Alignment and Attention. In *CVPR*, 5020–5029.

Liu, X.; Liang, D.; Yan, S.; Chen, D.; Qiao, Y.; and Yan, J. 2018. FOTS: Fast Oriented Text Spotting with a Unified Network. In *CVPR*, 5676–5685.

Long, S.; Ruan, J.; Zhang, W.; He, X.; Wu, W.; and Yao, C. 2018. Textsnake: A Flexible Representation for Detecting Text of Arbitrary Shapes. In *ECCV*, 19–35.

Lyu, P.; Liao, M.; Yao, C.; Wu, W.; and Bai, X. 2018. Mask Textspotter: An End-to-End Trainable Neural Network for Spotting Text with Arbitrary Shapes. In *ECCV*, 71–88.

Shi, B.; Wang, X.; Lyu, P.; Yao, C.; and Bai, X. 2016. Robust Scene Text Recognition with Automatic Rectification. In *CVPR*, 4168–4176.

Sun, Y.; Zhang, C.; Huang, Z.; Liu, J.; Han, J.; and Ding, E. 2018. TextNet: Irregular Text Reading from Images with an End-to-End Trainable Network. In *ACCV*.

Tian, Z.; Shu, M.; Lyu, P.; Li, R.; Zhou, C.; Shen, X.; and Jia, J. 2019. Learning Shape-Aware Embedding for Scene Text Detection. In *CVPR*.

Wang, W.; Xie, E.; Li, X.; Hou, W.; Lu, T.; Yu, G.; and Shao, S. 2019a. Shape Robust Text Detection With Progressive Scale Expansion Network. In *CVPR*.

Wang, X.; Jiang, Y.; Luo, Z.; Liu, C.-L.; Choi, H.; and Kim, S. 2019b. Arbitrary Shape Scene Text Detection With Adaptive Text Region Representation. In *CVPR*.

Xie, E.; Zang, Y.; Shao, S.; Yu, G.; Yao, C.; and Li, G. 2019. Scene Text Detection with Supervised Pyramid Context Network. In *AAAI*.

Xu, Y.; Wang, Y.; Zhou, W.; Wang, Y.; Yang, Z.; and Bai, X. 2019. Textfield: Learning a deep direction field for irregular scene text detection. *IEEE TIP*.

Zhang, Z.; Zhang, C.; Shen, W.; Yao, C.; Liu, W.; and Bai, X. 2016. Multi-Oriented Text Detection with Fully Convolutional Networks. In *CVPR*, 4159–4167.

Zhang, C.; Liang, B.; Huang, Z.; En, M.; Han, J.; Ding, E.; and Ding, X. 2019. Look More Than Once: An Accurate Detector for Text of Arbitrary Shapes. In *CVPR*.

Zhou, X.; Yao, C.; Wen, H.; Wang, Y.; Zhou, S.; He, W.; and Liang, J. 2017. EAST: An Efficient and Accurate Scene Text Detector. In *CVPR*, 2642–2651.