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

Liang Qiao,[1] Sanli Tang,[1] Zhanzhan Cheng,[2,1]* Yunlu Xu,[1] Yi Niu,[1] Shiliang Pu,[1] Fei Wu[2]

[1]Hikvision Research Institute, China; [2]Zhejiang University, China

{qiaoliang6, tangsanli, chengzhanzhan, xuyunlu, niuyi, pushiliang}@hikvision.com, wufei@cs.zju.edu.cn

Abstract

Many approaches have recently been proposed to detect irregular scene text and achieved promising results. However, their localization results may not well satisfy the following text recognition part mainly because of two reasons: 1) recognizing arbitrary shaped text is still a challenging task, and 2) prevalent non-trainable pipeline strategies between text detection and text recognition will lead to suboptimal performances. To handle this *incompatibility* problem, in this paper we propose an end-to-end trainable text spotting approach named Text Perceptron. Concretely, Text Perceptron first employs an efficient segmentation-based text detector that learns the latent text reading order and boundary information. Then a novel Shape Transform Module (*abbr.* STM) is designed to transform the detected feature regions into regular morphologies without extra parameters. It unites text detection and the following recognition part into a whole framework, and helps the whole network achieve global optimization. Experiments show that our method achieves competitive performance on two standard text benchmarks, i.e., ICDAR 2013 and ICDAR 2015, and also obviously outperforms existing methods on irregular text benchmarks SCUT-CTW1500 and Total-Text.

1 Introduction

Spotting scene text is a hot research topic due to its various applications such as invoice recognition and road sign reading in advanced driver assistance systems. With the advances of deep learning, many deep neural-network-based methods (Wang et al. 2012; Jaderberg, Vedaldi, and Zisserman 2014; Li, Wang, and Shen 2017; Liu et al. 2018; He et al. 2018) have been proposed for spotting text from a natural image, and have achieved promising results.

However, in the real-world, many texts appear in arbitrary layouts (*e.g.* multi-oriented or curved), which make quadrangle-based methods (Liao et al. 2017; Zhou et al. 2017; Zhang et al. 2018) cannot be well adapted in many situations. Some works (Dai et al. 2018; Long et al. 2018; Xie et al. 2019) began to focus on irregular text localization by segmenting text masks as detection results and achieved relatively good performance in terms of Intersection-over-Union (IoU) evaluation. However, they still leave many challenges to the following recognizing task. For example, a
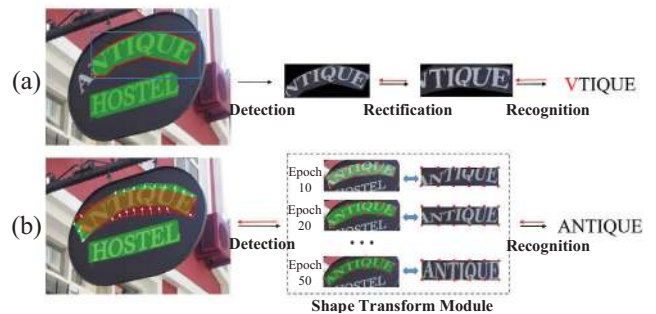
Figure 1: Illustration of the traditional pipelined text spotting process and Text Perceptron. Sub-figure (a) is a traditional pipeline strategy by combining text detection, rectification and recognition into a framework. Sub-figure (b) is an end-to-end trainable text spotting approach by applying the proposed STM. The black and red arrows mean the forward and backward processing, respectively. The red points denote generated fiducial points generated.

common pipeline of text spotting is to crop the masked texts within bounding-box regions, and then adopt a recognition model with rectification functions to generate final character sequences. Unfortunately, such strategy decreases the robustness of text spotting mainly in two aspects: 1) one needs to design extra rectification network, like methods in (Luo, Jin, and Sun 2019) and (Zhan and Lu 2019), to transform irregular texts into regular ones. In practice, it is hard to be optimized without human-labeled geometric ground truth, and also introduces extra computational cost. 2) Pipelined text spotting methods are not end-to-end trainable and result in suboptimal performance because the errors from the recognition model cannot be utilized for optimizing the text detector. In Figure 1(a), although the text detector provides true positive results, the clipped text masks still lead to wrong recognition results. We denote above problem *incompatibility* between text detection and recognition.

Recently, two methods were proposed for spotting irregular text in the end-to-end manners. (Lyu et al. 2018) proposed an end-to-end trainable network inspired by Mask-RCNN (He et al. 2017), aiming at reading irregular text character-by-character. However, this approach loses the

context information among characters, and also requires amounts of expenditure on character-level annotations. (Sun et al. 2018) attempted to transform irregular text with a perspective ROI module, but this operation has difficulty in handling some complicated distortions such as curved shapes.

These limitations motivate us to explore new and more effective method to spot irregular scene text. Inspired by (Shi et al. 2016), thin-plate splines (*abbr.* TPS) (Bookstein 1989) may be a feasible approach to rectify various-shaped text into regular form using a group of fiducial points. Although these points can be implicitly learned from cropped rectangular text by a deep spatial transform network (Jaderberg et al. 2015), the learning process of fiducial points is hard to be optimized. As a result, such methods are not robust especially for texts in some complex distortions.

In a more achievable way, we attempt to solve this problem as follows: 1) explicitly finding out a group of reliable fiducial points over text regions so that irregular text can be directly rectified by TPS, and 2) dynamically tuning fiducial points by back-propagating errors from recognition to detection. Specifically, we develop a Shape Transform Module (*abbr.* STM) to build a robust irregular text spotter and eliminate the *incompatibility* problem. STM integrates irregular text detection and recognition into an end-to-end trainable model, and iteratively adjusts fiducial points to satisfy the following recognition module. As shown in Figure 1(b), in the early training stage, despite high IoU in detection evaluation, the transformed text regions may not satisfy the recognition module. With end-to-end training, fiducial points will be gradually adjusted to obtain better recognition results.

In this paper, we propose an end-to-end trainable irregular text spotter named Text Perceptron which consists of three parts: 1) A segmentation-based detection module which orderly describes a text region as four subregions: the *center* region, *head, tail* and *top&bottom boundary* regions, detailed in Section 3. Here, boundary information not only helps separate text regions that are very close to each other, but also contributes to capture latent reading-orders. 2) STM for iteratively generating potential fiducial points and dynamically tuning their positions, which alleviates *incompatibility* between text detection and recognition. 3) A sequence-based recognition module for generating final character sequences.

Major contributions of this paper are listed as follows: 1) We design an efficient order-aware text detector to extract arbitrary-shaped text. 2) We develop the differentiable STM devoting to optimizing both detection and recognition in an end-to-end trainable manner. 3) Extensive experiments show that our method achieves competitive results on two regular text benchmarks, and also significantly surpasses previous methods on two irregular text benchmarks.

2   Related Works

Here, we briefly review the recent advances in text detection and end-to-end text spotting.

2.1   Text Detection

Methods of text detection can usually be divided into two categories: anchor-based methods and segmentation-based methods.

Anchor-based methods. These methods usually follows the technique of Faster R-CNN (Ren et al. 2015) or SSD (Liu et al. 2016) that uses anchors to provide rectangular region proposals. To overcome the significantly varying aspect ratios of texts, (Liao et al. 2017) designed long default boxes and filters to enhance text detection, and then (Liao, Shi, and Bai 2018) extended this work by generating quadrilateral boxes to fit the texts with perspective distortions. (Ma et al. 2018) proposed a rotated regional proposal network to enhance multi-oriented text detection. To detect arbitrary-shaped text, many Mask RCNN (He et al. 2017)-based methods, *e.g.*, CSE (Liu et al. 2019b), LOMO (Zhang et al. 2019) and SPCNet (Xie et al. 2019), were developed to capture irregular texts and achieved good performance.

Segmentation-based methods. These methods usually learn a global semantic segmentation without region proposals, which is more efficient compared to anchor-based methods. Segmentation can easily be used to describe text in arbitrary shapes but highly relies on complicated post-processes to separate different text instances. To solve this problem, (Wu and Natarajan 2017) introduced boundary semantic segmentation to reduce the efforts in post-proposing. EAST (Zhou et al. 2017) learned a shrink text region and directly regressed the multi-oriented quadrilateral boxes from text pixels. (Long et al. 2018) designed a series of overlapping disks with different radii and orientations to describe arbitrary-shaped text regions. (Wang et al. 2019) proposed a method that first generates text region masks with various shrinkage ratios and then uses a progressive expansion algorithm to produce the final text region masks. (Xu et al. 2019) predicted each text pixel and assigned them with a regression value denoting the direction to its nearest boundary to help separate different texts.

2.2   Text Spotting

Most of existing text-spotting methods (Liao, Shi, and Bai 2018; Liao et al. 2017; Wang et al. 2012) generally first localize each text with a trained detector such as (Zhou et al. 2017) and then recognize the cropped text region with a sequence decoder (Shi, Bai, and Yao 2017). For sufficiently exploiting the complementarity between detection and recognition, some works (He et al. 2018; Li, Wang, and Shen 2017; Liu et al. 2018) were proposed to jointly detect and recognize text instances in an end-to-end trainable manner, which utilized the recognition information to optimize the localization task. However, these methods are incapable of spotting arbitrary-shaped text due to the irrationality of rectangles or quadrangles. To address these problems, (Sun et al. 2018) adopted a perspective ROI transforming module to rectify perspective text, but this operation still has difficulty in handling serious curved text. (Lyu et al. 2018) proposed an end-to-end text spotter inspired by Mask-RCNN for detecting arbitrary-shaped text character-by-character, but this method loses the context information among characters and also requires character-level location annotations.
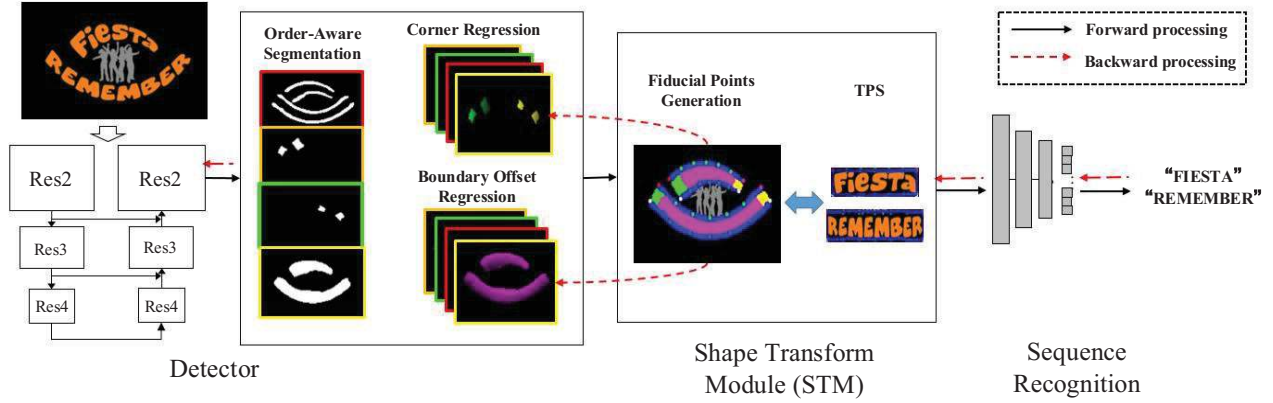
Figure 2: The workflow of Text Perceptron. The black and red arrows separately mean the forward and backward process.

3 Methodology

3.1 Overview

We propose a text spotter named Text Perceptron whose overall architecture is shown in Figure 2, which consists of three parts:

(1) The text detector adopts ResNet (He et al. 2016) and Feature Pyramid Network (*abbr.* FPN) (Lin et al. 2017) as backbone, and is implemented by simultaneously learning three tasks: an order-aware multiple-class semantic segmentation, a corner regression, and a boundary offset regression. In this way, the text detector can localize arbitrary-shaped text and achieve state of the art on text detection.

(2) STM is responsible for uniting text detection and recognition into an end-to-end trainable framework. This module iteratively generates fiducial points on text boundaries based on the predicted score and geometry maps, and then applies the differentiable TPS to rectify irregular text into regular form.

(3) The text recognizer is used to generate the predicted character sequences, which can be any traditional sequence-based method, such as CRNN (Shi, Bai, and Yao 2017), attention-based method (Cheng et al. 2017).

3.2 Text Detection Module

Order-aware Semantic Segmentation The text detector learns a global multi-class semantic segmentation, which is much more efficient than those Mask-RCNN-based methods. Inspired by (Xue, Lu, and Zhan 2018), we introduce text boundary segmentation to separate different text instances. Considering text with arbitrary shapes, we further category boundaries into *head*, *tail*, and *top&bottom boundary* types, respectively. In Figure 3, the green, yellow, blue and pink regions separately denote the *head*, *tail*, *top&bottom* boundaries and the *center* text region. Here, *head* and *tail* also capture potential information about text reading order (*e.g.* top to bottom for vertical text). Therefore, we learn the text detector by conducting the multi-class semantic segmentation task using several binary Dices Coefficient Loss (Milletari, Navab, and Ahmadi 2016) (denoted by $\mathcal{L}_{cls}$).

Corner and Boundary Regressions To boost the arbitrary-shaped segmentation performance as well as provide position information for fiducial points, we integrate two other regression tasks into the learning process, as shown in Figure 3 (c) and (d),

- *Corner Regression.* For pixels in *head* and *tail* regions, we regress the offsets (*e.g.* the $\Delta dx_1, \Delta dy_1, \Delta dx_2$ and Δdy_2) to their corresponding two corner points, which is denoted by $\mathcal{L}_{corner}$.

- *Boundary Offset Regression.* For pixels in *center* region, we regress the vertical and horizontal offsets to their nearest boundaries (*e.g.* the $\Delta dx'_1, \Delta dy'_1, \Delta dx'_2$ and $\Delta dy'_2$), which is denoted by $\mathcal{L}_{boundary}$.

Here, we adopt a proximity regression strategy to solve the inaccurate large-offset regression problem like in EAST (Zhou et al. 2017). That is, the Corner Regressions only regress their neighboring corresponding corners. In the Boundary Offset Regression, we can simply ignore or lower the loss weights of regression value generated from the larger side (*e.g.* $\Delta dx'_1, \Delta dx'_2$ for a horizontal text). In this way, our detector can well describe the texts with very large width-height ratios. Both of two regressions are trained with Smooth-L1 loss:

$$\mathcal{L}_{corner} \ or \ \mathcal{L}_{boundary} = \begin{cases} 0.5(\sigma z)^2 & |z| < 1/\sigma^2 \\ |z| - 0.5/\sigma^2 & \text{otherwise} \end{cases},$$
(1)

where z is the geometry offset value, and σ is a tunable parameter (default by 3).

The Detection Inference In the forward process, we generate predicted segmentation maps by orderly overlaying the segmented *center*, *head*, *tail*, and *top&bottom* boundary feature maps. Subsequently, text instances can be found as connected-regions of *center* pixels. We see that all text instances are easily separated by boundaries, and different *head* (or *tail*) regions will also be separated by *up&bottom* boundary region. Therefore, each *center* region can be matched with a neighboring pair of *head* and *tail* region during the pixel traversal process. Specifically, for text with
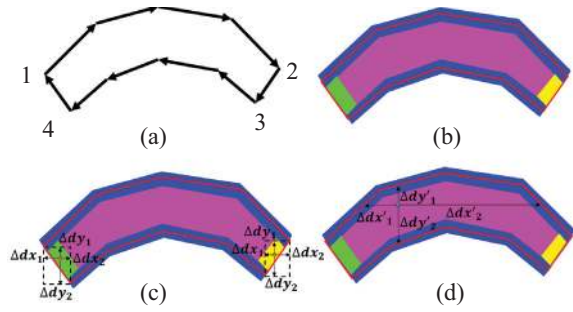
Figure 3: The label generation process.

Figure 4: The fiducial points generation process.

more than 1 *head* (or *tail*) regions, we choose the one with the maximum area as its *head* (or *tail*). While for predicted *center* text regions without corresponding *head* or *tail* region, we just treat them as false positives and filter them out.

Ground-Truth Generation   The process of ground-truth of segmentation and geometry map can be divided into three steps, as shown in Figure 3.

(1) Identifying four corners. We denote the 1st and 4th corners as the two corners in the *head* region, while the 2nd and 3rd corners are corresponding to the *tail* region, as shown in Figure 3(a). This weak-supervised information is not provided by most of the datasets, but we found that in general, polygon points $\{P'_1, ..., P'_M\}$ are usually annotated from the left-top corner to the left-bottom corner in a clockwise manner for text instances. Differently, for polygon annotations with a fixed number of points like SCUT-CTW1500 (Liu et al. 2019a), we can directly identify the four corner points by their indexes. However, for annotations with varying number of points like Total-Text (Ch'ng and Chan 2017), we can only obtain the 1st corner (P'_1) and 4th corner (P'_M). To search the 2nd and 3rd corners, we design a heuristic corner estimating strategy based on the assumptions that 1) two boundaries neighboring *tail* are nearly parallel, and 2) two neighbor interior angles of *tail* are closed to $\frac{\pi}{2}$. Therefore, the probable 2nd corner can be estimated as:

$$\arg\min_{P'_i}[\gamma(|\angle P'_i - \frac{\pi}{2}| + |\angle P'_{i+1} - \frac{\pi}{2}|) + |\angle P'_i + \angle P'_{i+1} - \pi|]$$
(2)

where $\angle P'_i$ is the degree of interior angle for polygon point P'_i, and γ is a weighting parameter (default by 0.5). Then the point P'_{i+1} following P'_i is treated as the 3-rd corner point. Specifically, for vertical text annotated from the top-left corner, we reassign its top-right corner as the 1st key corner.

(2) Generating score maps. Figure 3(b) shows the generated score maps. We firstly generate the *center* text regions follows by their annotations and then generate boundaries by referring to the shrink and expansion mechanism used in (Wu and Natarajan 2017). Differently, the *head* and *tail* score maps are generated by only applying the shrink operation, which submerges part of the *center* region. And *top&bottom* boundary region is then generated by applying both the expansion and shrink operations, which will partly submerge all of the other regions. In this way, we need
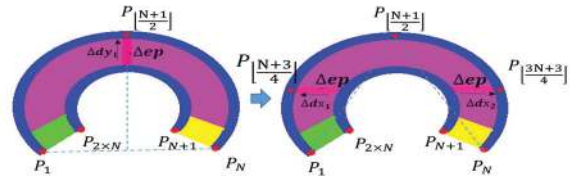
less effort on post-processing to separate different text instances and it is easy to match their relative *head* (or *tail*) region with a *center* region. Boundary widths are constrained as $\delta \times minLen$, where $minLen$ is the minimum length of edges in the text polygon and δ is a ratio parameter. Here, we set δ=0.2 for *top&bottom* boundaries and δ=0.3 for *head* and *tail*.

(3) Generating geometry maps. As mentioned in Corner and Boundary Regression, pixels belonging to the *head* region are assigned geometry offset values in 4 channels ($\Delta dx_1, \Delta dy_1, \Delta dx_2$ and Δdy_2) corresponding the 1st and 4th key corner, as shown in Figure 3(c). Similarly, the geometry map of the *tail* region is also formed in 4 channels. The geometry values of the *center* text region are computed as the horizontal and vertical offsets to the nearest boundaries, shown as $\Delta dx'_1, \Delta dy'_1, \Delta dx'_2$ and $\Delta dy'_2$ in Figure 3(d).

3.3   Shape Transform Module

STM is designed to iteratively generate initial fiducial points around text instances and transform text feature regions into regular shapes with the supervision of following recognition.

Fiducial Points Generation   With the learned segmentation maps and geometry maps, we propose to generate preset $2 \times N$ potential fiducial points ($N \geq 2$) for each text instance, denoted as $\{P_1, ..., P_N, P_{N+1}, ..., P_{2 \times N}\}$, which can be divided into two stages.

(1) Generating four corner points. We first obtain the positions of four corner fiducial points for each text feature region by averaging the coordinate of pixels with their predicted offsets in corresponding boundaries. Taking the 1-st corner point (P_1) as an example, it is computed based on all pixels in the *head* region $\mathcal{R}_{\mathcal{H}}$, and formalized by

$$P_1 = \left(\frac{\sum_{(x,y)\in\mathcal{R}_{\mathcal{H}}}(x + \Delta dx)}{||\mathcal{R}_{\mathcal{H}}||}, \frac{\sum_{(x,y)\in\mathcal{R}_{\mathcal{H}}}(y + \Delta dy)}{||\mathcal{R}_{\mathcal{H}}||}\right)$$
(3)

where $||.||$ means the number of pixels in $\mathcal{R}_{\mathcal{H}}$, and $\Delta dx, \Delta dy$ mean the predicted corner offsets corresponding to P_1. The other three corner points (P_N in $\mathcal{R}_{\mathcal{H}}$, P_{N+1}, $P_{2 \times N}$ in tail region $\mathcal{R}_{\mathcal{T}}$) can be calculated similarly.

(2) Generating other fiducial points. After obtaining four corner fiducial points, the other fiducial points can be located using a dichotomous method. This strategy is suitable for any arbitrary shaped text even serious curved or in different reading orders.

An example of the generation process is shown in Figure 4. We firstly connect P_1 and P_N, and judge whether the connected line has a longer span in horizontal direction or

vertical direction. Without loss generality, if it has a longer span in horizontal direction as shown, we calculate a middle point $P_{\lfloor (N+1)/2 \rfloor}$ between P_1 and P_N whose x-coordinate formed as:

$$x_{mid} = \frac{\lceil (N-1)/2 \rceil}{N-1} \times P_{1,x} + \frac{\lfloor (N-1)/2 \rfloor}{N-1} \times P_{N,x} \quad (4)$$

Then we use the learned boundary offsets from detector to predict the y-coordinate of $P_{\lfloor (N+1)/2 \rfloor}$. Concretely, we define the *band* region $\mathcal{B}_i$ as the part of the *center* region $\mathcal{R}_C$:

$$\mathcal{B}_{\lfloor \frac{1+N}{2} \rfloor} = \{(x,y) \in \mathcal{R}_C | x \in [x_{mid} - \Delta ep, x_{mid} + \Delta ep]\} \quad (5)$$

where Δep defines the range of the band region (default by 3). Similar to the generation of four corner fiducial points, we can use all pixels in the corresponding band region to predict an average y-coordinate for this fiducial point. Then, the coordinate of $P_{\lfloor (N+1)/2 \rfloor}$ can be formed as:

$$P_{\lfloor \frac{1+N}{2} \rfloor} = \left(x_{mid}, \frac{\sum_{(x_t,y_t) \in \mathcal{B}_{\lfloor \frac{1+N}{2} \rfloor}} y_t + \Delta dy'_t}{||\mathcal{B}_{\lfloor \frac{1+N}{2} \rfloor}||} \right) \quad (6)$$

where $\Delta dy'_t$ is the learned boundary offset value to the top-boundary ($\Delta dy'_1$). This process can be iteratively conducted using corresponding $\Delta dx'_t$ or $\Delta dy'_t$ until all of the fiducial points be calculated. Similarly, the fiducial points on the bottom boundary can be calculated by connecting P_{N+1} and $P_{2 \times N}$ and using the same strategy.

Shape Transformation   With the generated potential fiducial points on text boundaries, we can explicitly transform an irregular feature region $\mathcal{R}$ into a regular form $\mathcal{R}^*$. Here, fiducial points are mapped into some preset positions of the transformed feature map by directly applying TPS to the original feature regions. Specifically, we transform all feature regions into a region with width W and height H:

$$\mathcal{R}^* = TPS^{-1}(\mathcal{P}, \mathcal{R}), \quad (7)$$

where the fiducial point $P_i \in \mathcal{P}$ will be mapped into:

$$P_i^* = \begin{cases} \left((i-1) \times \frac{H - 2 \times \Delta w}{N-1} + \Delta w, \ \Delta h \right), 1 \leq i \leq N \\ \left((2 \times N - i) \times \frac{H - 2 \times \Delta w}{N-1} + \Delta w, \ H - \Delta h \right), \\ \qquad\qquad\qquad\qquad\qquad N < i \leq 2 \times N \end{cases} \quad (8)$$

where Δw and Δh are preset offsets (default by $0.1 \times W$ and $0.1 \times H$) to preserve space for fiducial points tuning.

Then, all text feature regions are packed into a batch and sent to the following recognition part. Here, we assume that the final predicted character strings Y are generated as:

$$Y = Recog(\mathcal{R}^*), \quad (9)$$

where '$Recog$' is the sequence recognition process.

Dynamically Finetuning Fiducial Points   The assumption here is that although text detector supervised by polygon annotations can generate satisfying polygon masks, the results may not always suitable for the following recognition.

To avoid the suboptimal problem and improve overall performance, Text Perceptron will back-propagate differences from '$Recog$' to each pixel value in $\mathcal{R}$ via STM, i.e.

$$\Delta \mathcal{R} = \frac{\partial Y}{\partial \mathcal{R}^*} \frac{\partial \mathcal{R}^*}{\partial \mathcal{R}}. \quad (10)$$

Then we can calculate the adjustment values of $\mathcal{P}$ by

$$\Delta \mathcal{P} = \frac{\partial Y}{\partial \mathcal{R}^*} \frac{\partial \mathcal{R}^*}{\partial \mathcal{R}} \frac{\partial \mathcal{R}}{\partial \mathcal{P}}. \quad (11)$$

Furthermore, we back-propagate $\Delta \mathcal{P}$ to the corresponding geometry maps in *head*, *tail* and *band* regions. Formally, for each pixel pi, we have

$$\Delta pi = \Delta \hat{pi} + \frac{\Delta \mathcal{P}}{||\mathcal{R}_{\mathcal{R}^*}||}, \quad (12)$$

where $\mathcal{R}_{\mathcal{R}^*} \in \{\mathcal{R}_H, \mathcal{R}_T, \mathcal{B}\}$ and $\Delta \hat{pi}$ is calculated from $\mathcal{L}_{corner}$ or $\mathcal{L}_{boundary}$.

3.4   End-to-End Training

Our recognition part can be implemented by any sequence-based recognition network, such as CRNN (Shi, Bai, and Yao 2017) or (Cheng et al. 2017).

The loss of the whole framework contains the following parts: the order-aware multi-class semantic segmentation, the corner regressions for pixels in *head* and *tail*, the boundary offset regression for pixels in the *center* region and the word recognition, that is,

$$\mathcal{L} = \mathcal{L}_{cls} + \lambda_b \mathcal{L}_{corner} + \lambda_c \mathcal{L}_{boundary} + \lambda_r \mathcal{L}_{recog}, \quad (13)$$

where λ_b, λ_c and λ_r are auto-tunable parameters, and $\mathcal{L}_{recog}$ is the loss from recognition.

Since learning fiducial points highly depends on the segmentation map learning, we use a soft loss weight strategy to automatically tune λ_b, λ_c and λ_r. In other words, in the first few epochs, fiducial points are mainly adjusted by regression tasks; while at the last few epochs, points are mainly restricted to recognition. Formally,

$$\lambda_b = \lambda_c = \lambda^* - \max(0.02 \times E, 0.5), \quad (14)$$

$$\lambda_r = \min(\max(-0.1 + 0.02 \times E, 0), \lambda_r^*), \quad (15)$$

where E is the number of training epochs, and λ^* and λ_r^* separately control the maximum loss weight of regression and recognition. In our experiments, we set $\lambda^* = 0.6$ and $\lambda_r^* = 0.8$.

4   Experiments

4.1   Datasets

The datasets used in this work are listed as follows:

SynthText 800k (Gupta, Vedaldi, and Zisserman 2016) contains 800k synthetic images that are generated by rendering synthetic text with natural images, and it is used as the pre-training dataset.

ICDAR2013 (Karatzas et al. 2013) (*abbr.* IC13) is collected as the focused scene text, which is mainly horizontal text containing 229 training images and 233 testing images.

Dataset	Method	Detection				End-to-End			Word Spotting		
		P	R	F	FPS	S	W	G	S	W	G
IC13	Textboxes (2017)	88.0	83.0	85.0	1.37	91.6	89.7	83.9	93.9	92.0	85.9
	Li et al. (2017)	91.4	80.5	85.6	-	91.1	89.8	84.6	94.2	92.4	88.2
	TextSpotter (2017)	-	-	-	-	89.0	86.0	77.0	92.0	89.0	81.0
	He et al. (2018)	91.0	88.0	90.0	-	91.0	89.0	86.0	93.0	92.0	87.0
	FOTS (2018)	-	-	88.2	**23.9**	88.8	87.1	80.8	92.7	90.7	83.5
	TextNet* (2018)	93.3	**89.4**	91.3	-	89.8	88.9	83.0	94.6	**94.5**	87.0
	Mask TextSpotter* (2018)	**95.0**	88.6	**91.7**	4.6	**92.2**	**91.1**	**86.5**	92.5	92.0	88.2
	Ours (2-stage)	92.7	88.7	90.7	10.3	90.8	90.0	84.4	93.7	93.1	86.2
	Ours (End-to-end)	94.7	88.9	**91.7**	10.3	91.4	90.7	85.8	**94.9**	94.0	**88.5**
IC15	EAST (2017)	83.6	73.5	78.2	**13.2**	-	-	-	-	-	-
	TextSnake* (2018)	84.9	80.4	82.6	1.1	-	-	-	-	-	-
	SPCNet* (2019)	88.7	85.8	87.2	-	-	-	-	-	-	-
	PSENet-1s* (2019)	86.9	84.5	85.7	1.6	-	-	-	-	-	-
	TextSpotter (2017)	-	-	-	-	54.0	51.0	47.0	58.0	53.0	51.0
	He et al. (2018)	87.0	**86.0**	87.0	-	**82.0**	**77.0**	63.0	**85.0**	**80.0**	65.0
	FOTS (2018)	91.0	85.2	**88.0**	7.8	81.1	75.9	60.8	84.7	79.3	63.3
	TextNet* (2018)	89.4	85.4	87.4	-	78.7	74.9	60.5	82.4	78.4	62.4
	Mask TextSpotter* (2018)	91.6	81.0	86.0	4.8	79.3	73.0	62.4	79.3	74.5	64.2
	Ours (2-stage)	91.6	81.8	86.4	8.8	78.2	74.5	63.0	80.6	76.6	65.5
	Ours (End-to-end)	**92.3**	82.5	87.1	8.8	80.5	76.6	**65.1**	84.1	79.4	**67.9**

Table 1: Results on IC13 and IC15. 'P', 'R' and 'F' separately mean the 'Precision', 'Recall' and 'F-Measure'. 'S', 'W' and 'G' mean recognition with strong, weak and generic lexicon, respectively. Superscript '*' means that the method considered the detection of irregular text.

ICDAR2015 (Karatzas et al. 2015) (*abbr.* IC15) is collected as incidental scene text consisting of many perspective text. It contains 1000 training and 500 testing images.

Total-Text (Ch'ng and Chan 2017) consists of multi-oriented and curve text and is therefore one of the important benchmarks in evaluating shape-robust text spotting tasks. It contains 1255 training and 300 testing images, and each text is annotated by a word-level polygon with transcription.

SCUT-CTW1500 (Liu et al. 2019a) (*abbr.* CTW1500) is a curved text benchmark consists of 1000 training and 500 testing images. In contrast to Total-Text, all text instances are annotated with 14-point polygons in the line-level.

4.2 Implementation Details

The detector uses ResNet-50 as the backbone and further be modified following the suggestions from (Huang et al. 2017) for obtaining dense features. We remove the fifth stage, modify $conv4\_1$ layer with stride=1 instead of 2, and apply atrous convolution for all subsequent layers to maintain enough receptive field. Training loss is calculated from the outputs of three stages: the fourth stage ($8\times$), the third stage ($8\times$), and the second stage ($4\times$) feature maps of FPN, and testing is only conducted on $4\times$ feature map. We directly adopt the attention-based network described in (Cheng et al. 2017) as the recognition model. All experiments are implemented in Caffe with 8 32GB-Tesla-V100 GPUs. The code will be published soon.

Data augmentation. We conduct data augmentation by simultaneously 1) randomly scaling the longer side of input images with length in range of [720, 1600], 2) randomly rotating the images with the degree in range of $[-15°, 15°]$, and 3) applying random brightness, jitters, and contrast on input images.

Training details. The networks are trained by SGD with batch-size=8, momentum=0.9 and weight-decay=5×10^{-4}. For both detection and recognition part, we separately pre-train them on SynthText for 5 epochs with initial learning rate 2×10^{-3}. Then, we jointly fine-tune the whole network using the soft loss weight strategy mention previously on each dataset for other 80 epochs. The initial learning rate is 1×10^{-3}. The learning rate will be divided by 10 for every 20 epochs. Online hard example mining (OHEM) (Shrivastava, Gupta, and Girshick 2016) strategy is also applied for balancing the foreground and background samples.

Testing details. We resize input images with the longer side 1440 for IC13, 2000 for IC15, 1350 for Total-text and 1250 for CTW1500. We set the number of fiducial points as 4 for two standard text datasets and 14 for two irregular text datasets. The detection results are given by connecting the predicted fiducial points. Note that, all images are tested in the single-scale.

4.3 Results on Standard Text Benchmarks

Evaluation on horizontal text. We first evaluate our method on IC13 mainly consisting of horizontal texts. Table 1 shows the results, and represents that our method achieve competitive performance compared to previous methods on the 'Detection', 'End-to-End' and 'Word Spotting' evaluation items. Besides, our method is also very efficient and achieves '10.3' of Frame Per Second (*abbr.* FPS).

Evaluation on perspective text. We evaluate our method on IC15 containing many perspective texts, and the results are shown in Table 1. In the detection stage, our method achieves comparable performance with the irregular text spotting methods such as TextNet and Mask TextSpotter. In the 'End-to-End' and 'Word Spotting' tasks, our method significantly outperforms previous irregular-text-based methods and achieves the remarkable state-of-the-art perfor-

mance on general lexicon cases, which demonstrates the effectiveness of our method.

4.4 Results on Irregular Text Benchmarks

Method	Detection			End-to-End	
	P	R	F	None	Full
TextSnake (2018)	82.7	74.5	78.4	-	-
FTSN (2018)	84.7	78.0	81.3		
TextField (2019)	81.2	79.9	80.6	-	-
SPCNet (2019)	83.0	82.8	82.9	-	-
CSE (2019b)	81.4	79.1	80.2	-	-
PSENet-1s (2019)	84.0	78.0	80.9	-	-
LOMO (2019)	75.7	**88.6**	81.6	-	-
Mask TextSpotter (2018)	69.0	55.0	61.3	52.9	71.8
TextNet (2018)	68.2	59.5	63.5	54.0	-
Ours (2-stage)	88.1	78.9	83.3	63.3	73.9
Ours (End-to-end)	**88.8**	81.8	**85.2**	**69.7**	**78.3**

Table 2: Result on Total-Text. "Full" indicates lexicons of all images are combined. "None" means lexicon-free.

We test our method on two irregular text benchmarks: Total-Text and CTW1500, as shown in Table 2 and 3. In the detection stage, our method outperforms all previous methods and surpasses the best result 2.3% on Total-Text and 2.4% on CTW1500 on F-measure evaluation.

Moreover, our method significantly outperforms previous methods on the precision item, which attributes to the false-positive filtering strategy. In the end-to-end case, our method significantly surpasses the best-reported results (Sun et al. 2018) by 15.7% on 'None' and the best of results (Lyu et al. 2018) by 6.5% on 'Full', which mainly attributes to STM achieving the end-to-end training strategies. Since CTW1500 releases the recognition annotation recently, there is no reported result on the end-to-end evaluation. Here, we report the end-to-end results lexicon-freely, and believe our method will significantly outperform previous methods.

Method	Detection			End-to-End
	P	R	F	None
TextSnake (2018)	69.7	85.3	75.6	-
TextField (2019)	83.0	79.8	81.4	-
CSE (2019b)	81.1	76.0	78.4	-
PSENet-1s (2019)	84.8	79.7	82.2	-
LOMO (2019)	69.6	**89.2**	78.4	-
Ours (2-stage)	**88.7**	78.2	83.1	48.6
Ours (End-to-end)	87.5	81.9	**84.6**	**57.0**

Table 3: Result on CTW1500. 'None' means lexicon-free.

In summary, the results on Total-Text and CTW1500 demonstrate the effectiveness of our method for arbitrary-shaped text spotting. Moreover, compared with 2-staged results, the end-to-end trainable strategy markedly boosts text spotting performance, especially for the recognition part.

4.5 Ablation Results of Fiducial Points

The number of fiducial points directly influences the detection and end-to-end results when texts are displayed in the curve or even waved shapes. Table 4 shows the result that how the number of fiducial points affects the detection and end-to-end evaluations on different benchmarks. It is clear that 4 points annotation is enough for regular benchmark

Dataset	Number of fiducial points							
	4	6	8	10	12	14	16	18
IC15	**87.1**	87.0	87.0	86.9	87.0	86.9	86.8	86.8
Total-Text	71.5	82.8	84.5	85.0	85.2	85.2	85.2	**85.3**
CTW1500	68.7	81.9	84.1	84.3	84.4	**84.6**	84.4	84.5
Total-Text	55.9	68.5	69.8	69.6	69.8	69.7	69.5	**69.9**
CTW1500	40.2	52.2	56.2	57.0	**57.1**	57.0	56.5	56.4

Table 4: Detection (top part) and end-to-end (bottom part) evaluation (F-measure) under varied number of fiducial points for different benchmarks.
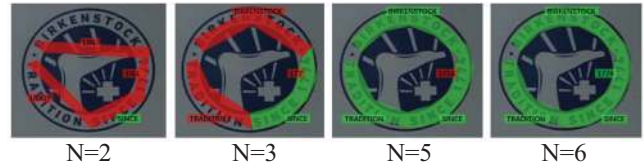

N=2   N=3   N=5   N=6

Figure 5: Results of Text Perceptron with different number of fiducial points (4,6,10,12).

such as IC15, and there is almost no influence on the result when the number of fiducial points increases. On the other hand, for two irregular benchmarks, the detection F-score as well as end-to-end F-score raises along with the increasing number of fiducial points, and the performance becomes stable when $2 \times N \geq 10$.

Figure 5 shows an example of end-to-end evaluation under different number of fiducial points. We see that the generated text masks by few fiducial points are hard to cover the entire curve texts. As the growing number of fiducial points, STM has more power to catch and rectify irregular text instances, which yields higher recognition accuracy.

In contrast to previous works, our method can generate any fixed number of fiducial points on text boundaries. The fiducial points generation method can also be used to annotate arbitrary-shaped text.

4.6 Visualization Results

Figure 6: Visualization results on origin images.

Figure 6 and Figure 7 demonstrate some visualization re-

Figure 7: 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.

Figure 8: Visualization of some failure samples.

sults in Total-Text and CTW1500 datasets. Text Perceptron shows its powerful ability in catching the reading order of irregular scene text (including curved, long perspective, vertical, etc.), and with the help of fiducial points which can further recognize text in a much simpler way. From the segmentation results, we find that 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 contain the different semantic information with that of the *center* region. Figure 6 also shows the visualization of some rectified irregular text instances, in which vertical texts can be well transformed into the "lying-down" shapes.

Failure Samples We illustrate some failure samples that are difficult for Text Perceptron, as shown in Figure 8.

Overlapped 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* text 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.

5 Conclusion

In this paper, we propose an end-to-end trainable text spotter named Text Perceptron aiming at spotting text with arbitrary-shapes. To achieve global optimization, a Shape Transform Module is proposed to unite the text detection and recognition into a whole framework. A segmentation-based detector is carefully designed to distinguish text instances and capture the latent information of text reading orders. Extensive experiments show that our method achieves competitive result in standard text benchmarks and the state-of-the-art in both detection and end-to-end evaluations on popular irregular text benchmarks.

References

Bookstein, F. L. 1989. Principal Warps: Thin-Plate Splines and the Decomposition of Deformations. *IEEE TPAMI* 11(6):567–585.

Bušta, M.; Neumann, L.; and Matas, J. 2017. Deep Textspotter: An End-to-End Trainable Scene Text Localization and Recognition Framework. In *ICCV*, 2223–2231.

Cheng, Z.; Bai, F.; Xu, Y.; Zheng, G.; Pu, S.; and Zhou, S. 2017. Focusing Attention: Towards Accurate Text Recognition in Natural Images. In *ICCV*, 5076–5084.

Ch'ng, C. K., and Chan, C. S. 2017. Total-text: A Comprehensive Dataset for Scene Text Detection and Recognition. In *ICDAR*, volume 1, 935–942.

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.

Gupta, A.; Vedaldi, A.; and Zisserman, A. 2016. Synthetic Data for Text Localisation in Natural Images. In *CVPR*, 2315–2324.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *CVPR*, 770–778.

He, K.; Gkioxari, G.; Dollar, P.; and Girshick, R. 2017. Mask R-CNN. In *ICCV*, 2980–2988.

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.

Huang, J.; Rathod, V.; Sun, C.; Zhu, M.; Korattikara, A.; Fathi, A.; Fischer, I.; Wojna, Z.; Song, Y.; Guadarrama, S.; et al. 2017. Speed Accuracy Trade-offs for Modern Convolutional Object Detectors. In *CVPR*, 7310–7311.

Jaderberg, M.; Simonyan, K.; Zisserman, A.; et al. 2015. Spatial Transformer Networks. In *NeurIPS*, 2017–2025.

Jaderberg, M.; Vedaldi, A.; and Zisserman, A. 2014. Deep Features for Text Spotting. In *ECCV*, 512–528.

Karatzas, D.; Shafait, F.; Uchida, S.; Iwamura, M.; i Bigorda, L. G.; Mestre, S. R.; Mas, J.; Mota, D. F.; Almazan, J. A.; and De Las Heras, L. P. 2013. ICDAR 2013 Robust Reading Competition. In *ICDAR*, 1484–1493.

Karatzas, D.; Gomez-Bigorda, L.; Nicolaou, A.; Ghosh, S.; Bagdanov, A.; Iwamura, M.; Matas, J.; Neumann, L.; Chandrasekhar, V. R.; Lu, S.; et al. 2015. ICDAR 2015 Competition on Robust Reading. In *ICDAR*, 1156–1160.

Li, H.; Wang, P.; and Shen, C. 2017. Towards End-to-end Text Spotting with Convolutional Recurrent Neural Networks. In *ICCV*, 5248–5256.

Liao, M.; Shi, B.; Bai, X.; Wang, X.; and Liu, W. 2017. TextBoxes: A Fast Text Detector with a Single Deep Neural Network. In *AAAI*, 4161–4167.

Liao, M.; Shi, B.; and Bai, X. 2018. TextBoxes++: A Single-Shot Oriented Scene Text Detector. *IEEE TIP* 27(8):3676–3690.

Lin, T.-Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; and Belongie, S. 2017. Feature Pyramid Networks for Object Detection. In *CVPR*, 2117–2125.

Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; and Berg, A. C. 2016. SSD: Single Shot Multibox Detector. In *ECCV*, 21–37. Springer.

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.

Liu, Y.; Jin, L.; Zhang, S.; Luo, C.; and Zhang, S. 2019a. Curved Scene Text Detection via Transverse and Longitudinal Sequence Connection. *PR* 90:337–345.

Liu, Z.; Lin, G.; Yang, S.; Liu, F.; Lin, W.; and Goh, W. L. 2019b. Towards Robust Curve Text Detection With Conditional Spatial Expansion. In *CVPR*.

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.

Luo, C.; Jin, L.; and Sun, Z. 2019. MORAN: A Multi-Object Rectified Attention Network for Scene Text Recognition. *PR*.

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.

Ma, J.; Shao, W.; Ye, H.; Wang, L.; Wang, H.; Zheng, Y.; and Xue, X. 2018. Arbitrary-Oriented Scene Text Detection via Rotation Proposals. *IEEE TMM* 20(11):3111–3122.

Milletari, F.; Navab, N.; and Ahmadi, S.-A. 2016. V-net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation. In *3DV*, 565–571.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *NeurIPS*, 91–99.

Shi, B.; Bai, X.; and Yao, C. 2017. An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition. *IEEE TPAMI* 39(11):2298–2304.

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

Shrivastava, A.; Gupta, A.; and Girshick, R. 2016. Training Region-based Object Detectors with Online Hard Example Mining. In *CVPR*, 761–769.

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*.

Wang, T.; Wu, D. J.; Coates, A.; and Ng, A. Y. 2012. End-to-End Text Recognition with Convolutional Neural Networks. In *ICPR*, 3304–3308.

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

Wu, Y., and Natarajan, P. 2017. Self-organized Text Detection with Minimal Post-processing via Border Learning. In *ICCV*, 5010–5019.

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*.

Xue, C.; Lu, S.; and Zhan, F. 2018. Accurate Scene Text Detection through Border Semantics Awareness and Bootstrapping. In *ECCV*, 370–387.

Zhan, F., and Lu, S. 2019. Esir: End-to-end scene text recognition via iterative image rectification. In *CVPR*, 2059–2068.

Zhang, S.; Liu, Y.; Jin, L.; and Luo, C. 2018. Feature Enhancement Network: A Refined Scene Text Detector. In *AAAI*.

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.