Symmetry-constrained Rectification Network for Scene Text Recognition

Mingkun Yang[1], Yushuo Guan[2], Minghui Liao[1], Xin He[4],
Kaigui Bian[2], Song Bai[3], Cong Yao[4] and Xiang Bai[1*]
[1]Huazhong University of Science and Technology,
[2]Peking University, [3]University of Oxford, [4]Megvii (Face++) Inc.
{yangmingkun, mhliao, xbai}@hust.edu.cn {david.guan, bkg}@pku.edu.cn
{yaocong2010, songbai.site, hexin7257}@gmail.com

Abstract

Reading text in the wild is a very challenging task due to the diversity of text instances and the complexity of natural scenes. Recently, the community has paid increasing attention to the problem of recognizing text instances of irregular shapes. One intuitive and effective solution to this problem is to rectify irregular text to a canonical form before recognition. However, these methods might struggle when dealing with highly curved or distorted text instances. To tackle this issue, we propose a Symmetry-constrained Rectification Network (ScRN) in this paper, based on the local attributes of text instances, such as center line, scale, and orientation. Such constraints with an accurate description of text shape enable ScRN to generate better rectification results than existing methods thus leading to higher recognition accuracy. Our method achieves state-of-the-art performance on text of both regular and irregular shapes. Specifically, the system outperforms existing algorithms by a large margin on datasets that contain quite a proportion of irregular text instances, e.g.*, ICDAR 2015, SVT-Perspective and CUTE80.*

1. Introduction

Scene text reading [60, 32, 59, 54, 53, 36, 35] is an important, active research area in computer vision, which can be applied to a wide range of real-world applications, such as self-driving cars, assistant position systems, and guide board recognition [43]. Scene text recognition, which aims at converting text regions in the images to machine-readable symbols, is a critical step in scene text reading systems. It remains challenging due to complex backgrounds, irregular shapes, varying fonts, non-uniform illuminations, *etc*.

Text instances in real-world scenarios have diverse shapes, *e.g.*, in horizontal, oriented, or curved forms. There have been a lot of works that focus on dealing with irregular text instances. AON [8] applies sequence recognition in four different orientations, which enables the recognition model to handle oriented text instances. RARE [45] and ASTER [46] employ a rectification module before recognition. The rectification modules can improve text recognition accuracy by rectifying text in irregular shapes into regular forms. These rectification modules are based on spatial transform network (STN) [21], which predicts the control points of the text outlines in a weakly supervised way, as shown in Fig. 1a. They can deal with the text of various shapes only with word-level supervision. Ideally, the control points should evenly spread along the upper and lower edges of the text region, and the paired upper and lower points should be symmetrical about the center line of text. However, these STN-based methods predict the control points separately and neglect the priors. Without any constraints for such priors, the rectification effect in highly curved or distorted occasions might be unsatisfactory.

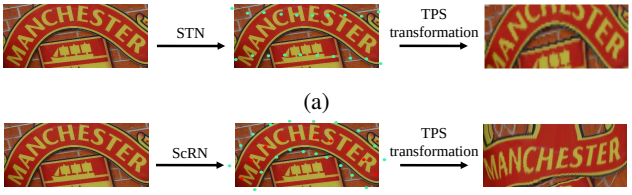

Figure 1: Comparison between ASTER [46] and ScRN (proposed in this paper), shown in (a) and (b) respectively.

To further improve the performance of irregular text rectification, we propose a Symmetry-constrained Rectification Network (ScRN) that uses the center line of each text instance and adds symmetrical constraints via some geometrical attributes, including the orientations of the text center line, the orientations and the scales of the characters. Specifically, each text center line is more flexible to describe the pose of either straight or curved text. Its associated geometrical attributes can reliably estimate the orientation and the boundary of text lines in vertical direction. Furthermore, the generation process of control points ensures the

*corresponding author.

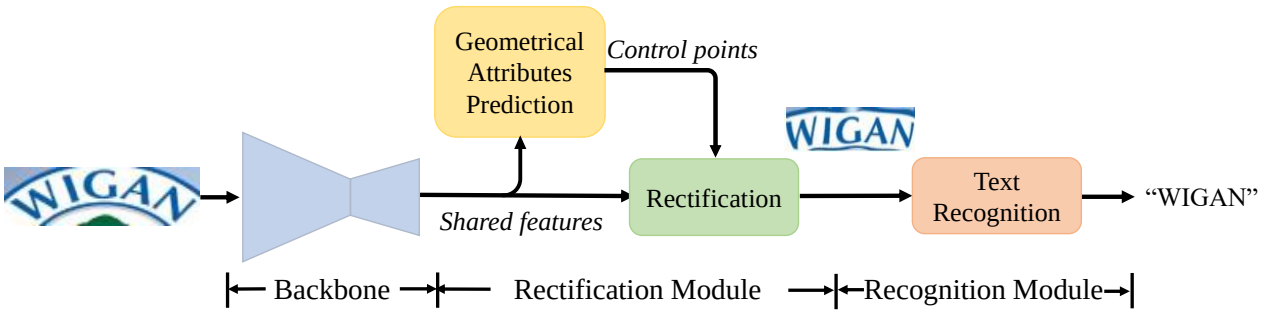

Figure 2: Pipeline of the proposed method.

symmetrical constraints in their spatial distribution. ScRN is a simple segmentation network which only consists of two convolutional layers. Therefore, it just incurs negligible computation and storage overhead when combined with a text recognizer. Compared with the previous STN-based rectification methods, ScRN has superiorities in both robustness and interpretability, profiting from its symmetric constraints. In this way, ScRN can further improve the text recognition accuracy by enhancing the performance of rectification on the irregular text, as illustrated in Fig. 1b.

The main contribution of this paper lies in the proposed rectification network for scene text recognition, whose advantages are three-fold. 1) The novel rectification network is more precise and robust, due to the elaborate description of text shape and the explicit symmetric constraints. 2) It is a simple and lightweight segmentation network, and thus the extra computation complexity is negligible when combined with existing text recognizers. 3) With the rectification network, we achieve state-of-the-art performance on the standard scene text recognition benchmarks.

2. Related Work

2.1. Text Recognition

Existing works on scene text recognition can be roughly divided into traditional and deep learning based methods.

A popular pipeline of traditional methods [10, 50, 49, 51, 38, 37, 39, 55, 4] is in the bottom-up architecture. They first localize every character with a character proposal extractor. Then, a character classifier is used to filter the proposals. Finally, the remained characters are grouped into words.

Deep learning based methods [20, 17, 25, 48, 45, 44, 7, 29, 3, 8, 46, 57] have been dominating this area in recent years. Jaderberg *et al.* [20] propose to take scene text recognition as a word classification problem by using a CNN classifier. However, it is limited to the pre-defined vocabulary. To overcome this limitation, various sequence-to-sequence models [44, 45, 7, 8, 46] are applied for scene text recognition, which do not rely on pre-defined vocabularies. These methods can be roughly divided into two subcategories by different sequence decoders. One subcategory is based on Connectionist Temporal Classification (CTC) [13, 44, 17] while the other is based on attention decoders [8, 46, 7]. More related papers are referred to [32].

2.2. Irregular Text Recognition

The irregular text includes, but is not limited to oriented or perspectively distorted text, curved text, *etc*. Recently, irregular text recognition [40, 52, 45, 8, 46, 30] becomes popular. Cheng *et al.* [8] encode the input image to four feature sequences of four directions to handle text of oriented shapes. Yang *et al.* [52] add character-level supervision to guide the attention learning on the 2D feature maps. Liu *et al.* [30] introduce "clean" images which contain no geometric deformation to supervise the learning process at both the pixel level and the feature level in a generative way. With such a generator-discriminator architecture, it can handle text on a curved path but fails in the text with a cluttered background. Shi *et al.* [45, 46] propose to add a rectification module before recognition. With only word-level supervision, they adopt the spatial transform network (STN) [21] to rectify the text in a weakly supervised manner. To improve the rectification results, Li *et al.* [26] bring extra supervision to STN and upgrade the model to a semi-supervised multi-task learning system, by labeling a portion of transformation parameters in the dataset. The control points are expected evenly spread along the upper and lower edges of text, and the paired upper and lower points should be symmetrical about the center line of text. Nevertheless, these rectification modules separately predict the control points and do not explicitly consider the constraints on the control points, which results in the limitations of their rectification effect. Our proposed method applies the constraints via geometrical attributes of text instances to rectify the irregular text, which gains both robustness and interpretability.

3. Our Method

As illustrated in Fig. 2, the proposed pipeline consists of three major parts: the shared backbone network, the rectification network and the recognition network. The model is end-to-end trainable and integrates the text rectification and recognition within a unified framework. The backbone is FPN [28] equipped with ResNet-50 [16], which is shared

by the rectification module and the recognition module. Using the shared feature maps, the rectification module yields dense pixel-wise predictions of text geometric attributes, with which the shared feature maps are expected to be rectified as regular ones via Thin-Plate-Spline (TPS) [6] transformation. Finally, the rectified feature maps are translated into a character sequence by the recognition module, where a shallow network is employed to convert the map to sequential features, followed by an attention decoder. The rectification module and recognition module are detailed in Sec. 3.1 and Sec. 3.2, respectively.

3.1. Rectification Module

The definition of the text shape is critical for text rectification because the rectification process can be considered as a shape transformation. Zhang *et al.* [58] design the text proposals according to symmetry while Long *et al.* [33] adopt local geometrical attributes, to represent the text shape for scene text detection. From the above analysis in Sec. 1, we conclude the symmetrical constraints are necessary for precise text rectification. To add such constraints into our rectification module, we use text center line with its associated geometrical attributes, such as character orientation, text orientation and text scale to describe the shape of a text instance. In this section, we first introduce a new representation for text rectification. Then we describe how to rectify text images with the given geometric attributes. At last, we highlight the necessity to introduce the character orientation for accurate rectification.

3.1.1 Definition

The geometrical attributes of text for rectification are illustrated in Fig. 3, including the text center line, the scale s, the character orientation φ and the text orientation θ.

A text instance can be viewed as an ordered character sequence $A = \{A_1, ..., A_i, ..., A_m\}$, where m is the number of characters. Each character A_i has a bounding box B_i, which is annotated with a free-form quadrilateral. First, we construct a center point list $C = \{c_{head}, c_1, ..., c_i, ..., c_m, c_{tail}\}$, which consists of the center point c_i of each B_i as well as the midpoint of B_1's left edge c_{head} and the midpoint of B_m's right edge c_{tail}. Then the text center line (TCL) is constructed by linking the center points in sequential order. Each center point is associated with a group of geometrical attributes, *i.e.*, $geo_i = (c_i; s_i; \varphi_i; \theta_i)$, where s_i is the scale, φ_i is the character orientation and θ_i is the text orientation. Specifically, the scale s_i is half the height of the character. The text orientation θ_i is defined as the tangential direction of $c_i \rightarrow c_{i+1}$. The character orientation φ_i is defined as the direction from the midpoint of the top edge to the midpoint of the bottom edge. For the points on the TCL but not in C, the values

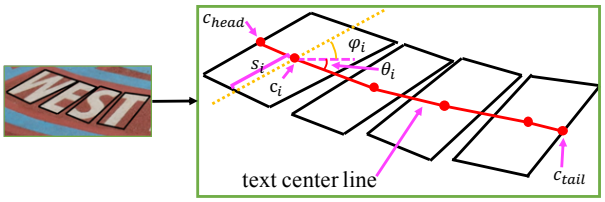

Figure 3: Illustration of the text representation.

of their geometrical attributes are linearly interpolated with two nearest center points. In this way, the shape of the text instance is precisely described and can be leveraged for the subsequent rectification step.

3.1.2 Geometrical Attributes Prediction

The rectification process is shown in Fig. 4. To yield the geometrical attributes, we employ a lightweight predictor which only consists of two convolutional layers. The output of this predictor is $F = \{f_1, f_2, ..., f_6\}$. f_1 represents the probability of pixels on the TCL, f_2 represents the character scale s at each pixel, f_3, f_4, f_5, and f_6 are pixel-wise predictions for $\cos\theta$, $\sin\theta$, $\cos\varphi$ and $\sin\varphi$, respectively. Specifically, $\cos\varphi$ and $\sin\varphi$ are normalized to ensure that their quadratic sum equals to 1, as depicted in Eqn. (1). $\cos\theta$ and $\sin\theta$ are normalized in the same way. After that, TCL score map, s, $\cos\theta$, and $\sin\theta$ are used to extract the central point list C, whose length is variable. More details about this process are referred to [33]. Then, C, s, $\cos\varphi$, and $\sin\varphi$ are used for rectification.

$$\cos\varphi = \frac{f_5}{\sqrt{{f_5}^2 + {f_6}^2}}, \quad \sin\varphi = \frac{f_6}{\sqrt{{f_5}^2 + {f_6}^2}}. \tag{1}$$

3.1.3 Rectification

Thin-Plate-Spline (TPS) transformation is employed to rectify the shared feature maps M to regular ones M_r. In order to compute the TPS transformation $\mathbf{T}$, we need to generate a pair of point sets $P = \{p_1, ..., p_i, ..., p_{2k}\}$ and P', which represent the fiducial points in the irregular feature maps and the predefined anchor points on the M_r, respectively. The procedure is given in Fig. 4. First, we equidistantly sample k points from C, named $\overline{C} = \{\overline{c}_1, ..., \overline{c}_i, ..., \overline{c}_k\}$. For each $\overline{c}_i$, we take two points at a distance s_i along the character orientation, which is expressed in $(\cos\varphi_i, \sin\varphi_i)$. The coordinates of the two points are computed via

$$\begin{aligned} p_{2i-1} &= \overline{c}_i + (s_i \times \cos\varphi_i, -s_i \times \sin\varphi_i), \\ p_{2i} &= \overline{c}_i + (-s_i \times \cos\varphi_i, s_i \times \sin\varphi_i). \end{aligned} \tag{2}$$

P' is evenly placed along the top and bottom borders of the regular feature maps. Given P and P', the transformation matrix $\mathbf{T}$ is calculated. Then, we apply $\mathbf{T}$ to every pixel locations in M_r and obtain a sampling grid on M, with which, M_r is sampled from M using bilinear interpolation.

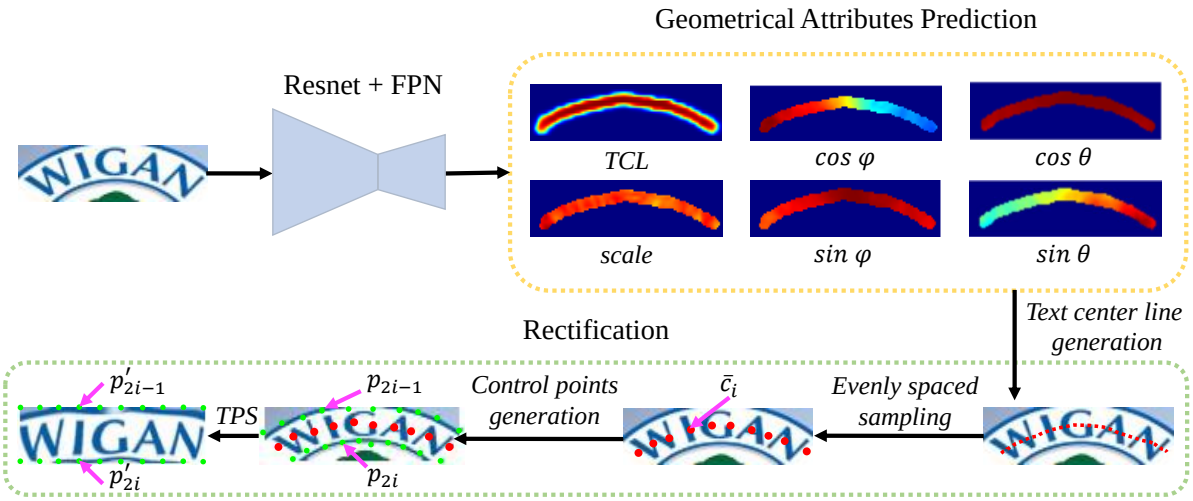

Figure 4: The rectification process. Note that, for all figures in this paper, we use the input image to illustrate these points and rectified results, but the rectification is actually operated on the shared feature maps.

Theoretically, TPS transformation is able to handle variable-size fiducial points, and thus C can be directly used to obtain the fiducial points P. However, to build a mini-batch for batch-wise training, the length of P should be predefined and fixed. Therefore, we resample the central point list C to obtain $\overline{C}$ with a fixed length.

3.1.4 Character Orientation

When bounding boxes of all characters are rectangular, the character orientation is perpendicular to the text orientation. However, in more general cases, the orientation perpendicular to the text orientation is not the correct character orientation, which may lead to a failed rectification. As illustrated in Fig. 5, when the normal direction of the center line is not the same as the character orientation, the rectification based on the character orientation φ is much better than the other one. So it is necessary to add the character orientation φ into the text geometric attributes for text rectification.

3.2. Recognition Module

The text recognition module aims to predict a character sequence from the rectified shared features. Using the hierarchical structure of the shared backbone, we obtain an enriched feature map. We use a sub-network to further encode the map to vector sequence before being fed into the final attention decoder. The settings of the recognition module are detailed in Tab. 1.

To reserve more discriminable features of the characters in compact text or in narrow shapes, the input feature is reduced only once along the width axis while keeps collapsing along the height axis until it reduces to 1. Then, the feature map is converted into a feature sequence by 1-stride sliced along the width axis. Finally, a Bidirectional

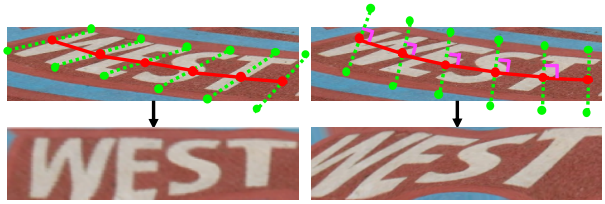

Figure 5: Control points and rectification results using the character orientation (Left) and normal direction of text orientation (Right).

LSTM [14] is attached to capture the long-range dependencies in both directions, resulting in a higher-level feature sequence $\mathbf{H} = \{h_1, ..., h_n\}$, where n is the length of $\mathbf{H}$.

Next, the common attention-based decoder [2] equipped with GRU [9] is adopted to translate the feature sequence $\mathbf{H}$ into a symbol sequence $\mathbf{y} = \{y_1, ..., y_T\}$, where T is the number of characters. To generate sequences of variable lengths, a special end-of-sequence symbol (EOS) is inserted at the end of the target sequence. The decoder iteratively predicts a symbol y_t at step t until EOS is emitted.

Given the input image I, the recognition loss can be formulated as

$$L_{\text{recog}} = -\frac{1}{T}\sum_{t=1}^{T}\log p(y_t|I). \tag{3}$$

3.3. Training and Inference

3.3.1 Training Objective

We add explicit supervision into both the rectification module and the recognition module. The whole network is trained end-to-end, with the following objective function,

$$L = \mathbf{1}_{I\in\text{SynthText}}(L_{\text{geo}}) + L_{\text{recog}}. \tag{4}$$

For an input image I, the loss is comprised of two parts,

Layer Name	Configuration	Out Size
conv1_x	$3 \times 3, 1 \times 1, 1 \times 1, 64$	16×64
	$3 \times 3, 1 \times 1, 1 \times 1, 64$	
conv2_x	maxpool:$2 \times 2, 2 \times 2, 0 \times 0$	8×32
	$3 \times 3, 1 \times 1, 1 \times 1, 128$	
	$3 \times 3, 1 \times 1, 1 \times 1, 128$	
conv3_x	maxpool:$2 \times 1, 2 \times 1, 0 \times 0$	4×32
	$3 \times 3, 1 \times 1, 1 \times 1, 256$	
	$3 \times 3, 1 \times 1, 1 \times 1, 256$	
conv4_x	maxpool:$2 \times 1, 2 \times 1, 0 \times 0$	1×31
	$2 \times 2, 1 \times 1, 0 \times 0, 256$	
Bi-LSTM	256	31
fc	nc	nc

Table 1: The architecture of recognition module. The configuration has the following format: $\{kernel_h \times kernel_w, stride_h \times stride_w, pad_h \times pad_w, channels\}$ for convolutional layers and maxpooling layers, {dimensions} for the number of features in the LSTM hidden state or fully-connected layers. "out size" is the feature map size of convolutional layers or the sequence length of recurrent layer. "nc" is the number of symbols.

as shown in Eqn. (4). L_{geo} measures the deviation of the predicted geometrical attributes with the ground truth. We train our model with SynthText [15] and Synth90k [18]. Synth90k has no annotations of char-level or word-level bounding boxes, so it is not used to supervise the training of geometrical attributes prediction.

$$\begin{aligned} L_{\text{geo}} =& \lambda_1 L_{tcl} + \lambda_2 L_s + \lambda_3 L_{sin\theta} \\ &+ \lambda_4 L_{cos\theta} + \lambda_5 L_{sin\varphi} + \lambda_6 L_{cos\varphi}, \end{aligned} \tag{5}$$

where L_{tcl} is cross-entropy loss for TCL. L_s, $L_{sin\theta}$, $L_{cos\theta}$, $L_{sin\varphi}$, and $L_{cos\varphi}$ are calculated as Smoothed-L1 loss [11],

$$\begin{pmatrix} L_s \\ L_{sin\theta} \\ L_{cos\theta} \\ L_{sin\varphi} \\ L_{cos\varphi} \end{pmatrix} = SmoothedL1 \begin{pmatrix} \frac{\widehat{s}-s}{s} \\ \widehat{sin\theta} - sin\theta \\ \widehat{cos\theta} - cos\theta \\ \widehat{sin\varphi} - sin\varphi \\ \widehat{cos\varphi} - cos\varphi \end{pmatrix}, \tag{6}$$

where $\widehat{s}$, $\widehat{sin\theta}$, $\widehat{cos\theta}$, $\widehat{sin\varphi}$ and $\widehat{cos\varphi}$ are the predicted values, while s, $sin\theta$, $cos\theta$, $sin\varphi$ and $cos\varphi$ are their ground truth correspondingly. L_{geo} for pixels outside the TCL is set to 0, since the geometrical attributes make no sense to non-TCL points.

The hyper-parameters, λ_1, λ_2, λ_3, λ_4, λ_5, and λ_6 are all set to 1 in our experiments.

3.3.2 Training Strategy

The feature maps generated from the backbone are shared by both the rectification module and the recognition module. Our training strategy is two-staged. In the first stage, the shared features are rectified with the ground truth geometrical attributes. Then, the rectified features are used for the recognition module training. Since Synth90k is not annotated with geometrical attributes, the shared features from Synth90k are not rectified in this stage. In the second stage, we use the predicted geometrical attributes for rectification. In this stage, all shared features are rectified before being fed into the recognition module.

4. Experiments

4.1. Datasets

We evaluate our method on 4 general benchmarks, which mainly consist of regular text instances and 3 datasets with irregular text instances, to demonstrate its rectification ability on curved, distorted and oriented text. A brief description of these datasets is as follows.

IIIT5K-Words (IIIT5K) [37] contains 3,000 web images for testing. Each image is associated with a 50-word lexicon and a 1k-word lexicon.

Street View Text (SVT) [49] consists of 647 testing images, which are collected from the Google Street View. Many images are heavily corrupted by noise, blur or in low resolution. Each image specifies a 50-word lexicon.

ICDAR 2003 (IC03) [34] contains 860 images of cropped words after filtering. Following Wang *et al.* [49], words with non-alphanumeric characters or less than three characters are discarded. Each image has a 50-word lexicon and a "full lexicon" which contains all lexicon words.

ICDAR 2013 (IC13) [24] inherits most of its data from IC03 and contains 1,015 cropped word images.

ICDAR 2015 (IC15) [23] is collected via a pair of Google Glasses without careful positioning and focusing. The dataset contains 2,077 images with various distortions.

SVT-Perspective (SVTP) [40] is specifically proposed to evaluate the performance of perspective text recognition algorithms. It consists of 645 images for testing.

CUTE80 (CUTE) [41] is designed to evaluate curved text recognition. It has 288 cropped images for testing.

4.2. Implementation Details

The proposed method is implemented in PyTorch. Images are resized to 64×256 before being fed into the network. The resolutions of feature maps produced by the shared backbone and the rectified feature maps are both $1/4$ size of the input image, namely 16×64. Accordingly, the size of ground truth maps F is also 16×64. We expand one more pixel around TCL since a single-point line is prone to noise. The geometrical attributes on the expanded points keep the same with the nearest point on the original TCL. To apply TPS transformation in the mini-batch, we equidistantly sample k=10 points after $\hat{C}$ is extracted. In

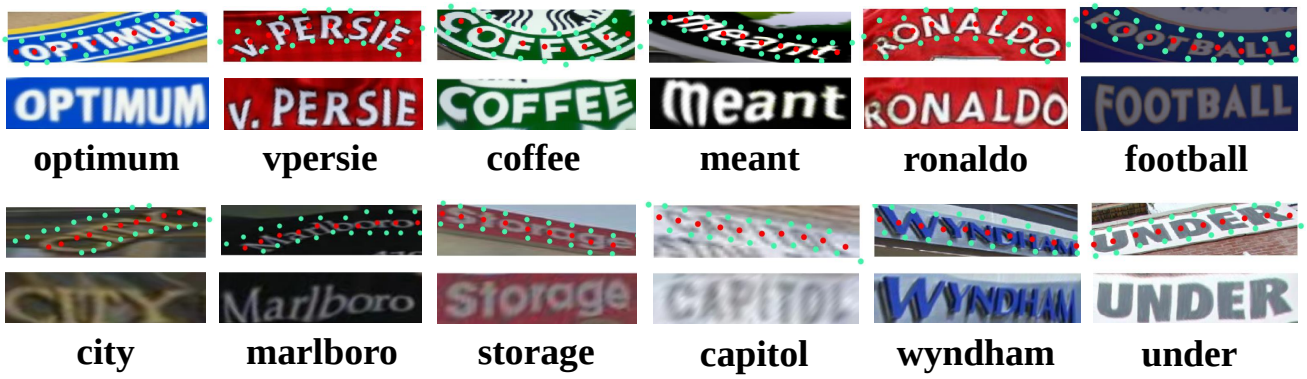

Figure 6: Selected results from SVTP and CUTE80, which suffer from severe distortion. For every three rows, the first row shows the input image with evenly sampled center points (visualized as red points) and green control points. The second row shows the rectified images. The last row is the recognition results.

total, 95 symbols are recognized, including digits, upper-case and lower-case letters, 32 punctuation marks and an end-of-sequence symbol (EOS).

Our model is trained on SynthText and Synth90k from scratch. We adopt the ADADELTA [56] with default hyper-parameters (rho=0.9, eps=1e-6, weight_decay=0) to minimize the objective function. Each mini-batch has 512 samples which are randomly selected from the two datasets. As mentioned in Sec. 3.3.2, our model is trained in two stages. In the first stage, we set the initial learning rate to 1.0 and decay it to 0.1 and 0.01 at the 4th epoch and the 5th epoch. The first stage finishes in the 6th epoch. In the second stage, the predicted geometrical attributes are used for rectification, and the model is trained for another epoch. All models are trained on 4 NVIDIA TITAN Xp graphics cards.

4.3. Effect of Rectification

To analyze the effect of rectification, we remove the rectification module in our pipeline as the baseline, where the feature maps generated from the backbone are fed into the recognition module directly. As shown in Tab. 3, the model with the rectification module outperforms the baseline nearly on all datasets, particularly on IC15 (+1.8%), SVTP (+3.1%) and CUTE80 (+3.1%). The significant improvements on three irregular text datasets demonstrate the effectiveness of the proposed ScRN. Furthermore, the attached rectification module only needs negligible computation and storage overhead, since it only consists of two convolutional layers and some simple postprocessing. Specifically, the baseline model and our method spend 12ms and 13ms in the inference stage, respectively.

To further explain the improvements, we visualize several images with different types of distortions to illustrate the rectification results in Fig. 6. With the proposed geometrical attributes, ScRN can obtain a precise description of the text shape, which finally results in evenly placed control points along the top and bottom text edges. Therefore, the followed TPS transformation can easily rectify these irregular text images. Although some rectified images are still slightly distorted, the images become more readable than the original ones and can be correctly recognized.

Variants	IIIT5k	SVT	IC03	IC13	IC15	SVTP	CUTE
baseline	88.4	79.9	92.1	88.9	67.3	66.5	80.6
multi-loss	87.6	79.1	91.3	90.0	67.0	66.7	79.5
ours	88.5	81.3	91.2	90.0	68.8	68.2	81.9

Table 2: Recognition accuracy to explore the effect of rectification module. All models are trained on SynthText only.

Unlike ASTER, our model is end-to-end trained with both the recognition loss and geometry loss. To make it clear whether the extra geometry loss or the rectification module improve the recognition results, we study another variant of the proposed model called multi-loss baseline, in which geometry loss is retained but the rectification module is discarded. In this part, the baseline, multi-loss baseline and our method are trained on SynthText only. Their performance is given in Tab. 2. Compared to the baseline model, the multi-loss variant achieves comparable results while our method obtains improvements on most datasets, except a slight decrease on IC03. These results reveal that the improvements are derived from the rectification module, rather than the extra geometry loss.

4.4. Comparison with STN-based Methods

In this section, we compare our method with two STN-based methods. One is ASTER, a well-known STN-based method. The other one is similar with ASTER, but extra supervision is injected for STN. But we do not compare our method with ASTER directly here, since our method rectifies shared feature maps instead of raw images, considering complexity and efficiency. Therefore, we build another STN-based model, namely STN_baseline, for a fair comparison. STN_baseline shares the same backbone and recognition module with our method. It only replaces our

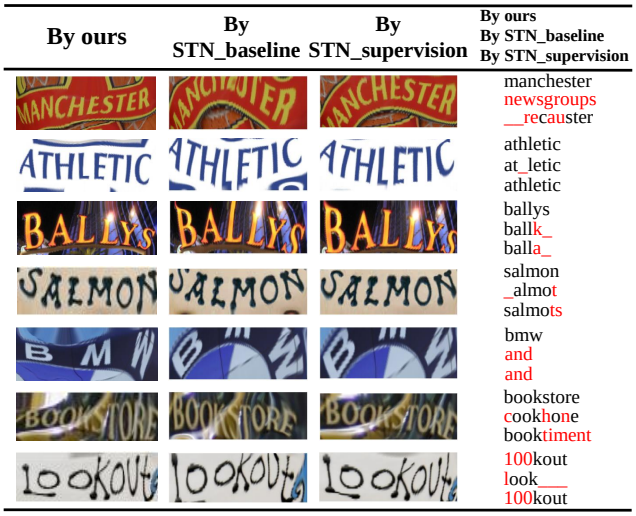

Figure 7: Rectified results produced by our proposed ScRN, STN_baseline and STN_supervision, as well as their corresponding recognition results. Red characters are mistakenly recognized characters. Underlines in red represent the missed characters.

rectification module with an STN network, which has a similar architecture with the rectification network of ASTER. The other STN-based method has the same structure as STN_baseline. The only difference is the extra supervision to further improve the accuracy of the predicted control points. This variant derives from [26] and we name it STN_supervision. All methods share the same training strategy. The results are shown in Tab. 3. Overall, STN_baseline outperforms the baseline model, meanwhile performs slightly worse than the STN_supervision. The conclusion is consistent with ASTER and [26]. Then we detail the comparisons with our method as follows.

On the datasets with irregular text such as IC15, SVTP and CUTE80, our method outperforms STN_baseline with improvements of 0.5%, 1.4% and 1.7%, respectively. With the extra supervision to STN, STN_supervision exceeds STN_baseline slightly but still performs worse than our method by 0.2%, 1.1%, 1.0% on IC15, SVTP and CUTE80, respectively. Profiting from the elaborate description of text pose, the rectification is more robust and accurate. In Fig. 7, we show some rectified results yielded by the three methods. Overall, STN-based methods suffer from heavily curved cases and predict imprecise control points, which lead to wrong rectifications while our method works well. Although our method fails to perfectly rectify text images with messy background and text with rare fonts, it can obtain more readable results.

The results reveal that the geometrical attributes are more helpful than the weakly supervised network and the simple supervised network for control points generation. Besides, the prediction network for geometrical attributes is much smaller and only trained with synthetic data, which is efficient and inexpensive.

Variants	IIIT5k	SVT	IC03	IC13	IC15	SVTP	CUTE
baseline	94.4	86.9	94.7	93.6	76.9	77.7	84.4
STN_baseline	94.1	87.6	95.0	93.2	78.2	79.4	85.8
STN_supervision	94.0	88.1	94.9	93.8	78.5	79.7	86.5
ScRN	94.4	88.9	95.0	93.9	78.7	80.8	87.5
ScRN*	95.0	88.4	95.6	93.7	78.4	81.1	90.6

Table 3: Recognition accuracy of different variants.

We also study a variant of our model, named ScRN* in Tab. 3 where we apply the rectification module to the input image, rather than the shared feature maps. In this variant, the backbone network is repeated twice without sharing parameters. So the elapsed time and the model size are nearly doubled. Compared with this variant, our method achieves comparable or even better results while avoiding the heavy computation and space cost.

4.5. Comparison with State of the Art

We also compare our method with previous state-of-the-art models. Tab. 4 summarizes the recognition results on seven text recognition datasets. The datasets IIIT5k, SVT and IC03 have lexicons to constrain recognition results. When analyzing the recognition accuracy of different models on these datasets, the predicted word will be replaced by the lexicon word that has the least edit distance with the original prediction. We achieve 6 best results out of 12, compared with other state-of-the-art methods.

Our method works effectively on datasets containing irregular text. Especially, we get an 8% improvement on CUTE80 compared with ASTER. We also outperform other state-of-the-art methods on SVTP and IC15 by 2.3% and 2.6%, respectively. The improvement gives credit to our rectification module, which attenuates text irregularities and therefore decreases the recognition difficulty. Compared with AON [8], our method provides a more intuitive way to represent text directions. Recur to the symmetrical constraints brought by the geometrical attributes, our method obtains more precise control points compared with ASTER.

Although our method mainly targets at irregular text recognition, it also achieves comparable or even better performance on regular datasets. Compared with ASTER, we get respectively 1%, 0.5%, and 2.1% improvements on IIIT5K, IC03, and IC13 with no lexicon. On SVT, our method performs slightly worse than ASTER by 0.6%. We conjecture that it is because the images in SVT always contain some incomplete characters on the left side. A unidirectional attention decoder in the left-to-right order suffers from the noise while the bidirectional one in ASTER can alleviate this effect.

4.6. Limitations

We also illustrate some failure cases produced by ScRN in Fig. 8. In Fig. 8a, several characters are incorrectly rec-

Methods	IIIT5k			SVT		IC03			IC13	IC15	SVTP	CUTE80
	50	1k	0	50	0	50	Full	0	0	0	0	0
Wang *et al.* [49]	-	-	-	57.0	-	76.0	62.0	-	-	-	-	-
Mishra *et al.* [38]	64.1	57.5	-	73.2	-	81.8	67.8	-	-	-	-	-
Wang *et al.* [51]	-	-	-	70.0	-	90.0	84.0	-	-	-	-	-
Bissacco *et al.* [5]	-	-	-	-	-	90.4	78.0	-	87.6	-	-	-
Almazan *et al.* [1]	91.2	82.1	-	89.2	-	-	-	-	-	-	-	-
Yao *et al.* [55]	80.2	69.3	-	75.9	-	88.5	80.3	-	-	-	-	-
Rodríguez-Serrano *et al.* [42]	76.1	57.4	-	70.0	-	-	-	-	-	-	-	-
Jaderberg *et al.* [22]	-	-	-	86.1	-	96.2	91.5	-	-	-	-	-
Su and Lu [47]	-	-	-	83.0	-	92.0	82.0	-	-	-	-	-
Gordo [12]	93.3	86.6	-	91.8	-	-	-	-	-	-	-	-
Jaderberg *et al.* [20]	97.1	92.7	-	95.4	80.7	98.7	**98.6**	93.1	90.8	-	-	-
Jaderberg *et al.* [19]	95.5	89.6	-	93.2	71.7	97.8	97.0	89.6	81.8	-	-	-
Shi *et al.* [44]	97.8	95.0	81.2	97.5	82.7	98.7	98.0	91.9	89.6	-	-	-
Shi *et al.* [45]	96.2	93.8	81.9	95.5	81.9	98.3	96.2	90.1	88.6	-	71.8	59.2
Lee *et al.* [25]	96.8	94.4	78.4	96.3	80.7	97.9	97.0	88.7	90.0	-	-	-
Yang *et al.* [52]	97.8	96.1	-	95.2	-	97.7	-	-	-	-	75.8	69.3
Cheng *et al.* [7]	99.3	97.5	87.4	97.1	85.9	**99.2**	97.3	94.2	93.3	70.6	-	-
Cheng *et al.* [8]	99.6	98.1	87.0	96.0	82.8	98.5	97.1	91.5	-	68.2	73.0	76.8
Liu *et al.* [29]	-	-	92.0	-	85.5	-	-	92.0	91.1	74.2	78.9	-
Bai *et al.* [3]	99.5	97.9	88.3	96.6	87.5	98.7	97.9	94.6	**94.4**	73.9	-	-
Liu *et al.* [31]	97.0	94.1	87.0	95.2	-	98.8	97.9	93.1	92.9	-	-	-
Liu *et al.* [30]	97.3	96.1	89.4	96.8	87.1	98.1	97.5	94.7	94.0	-	73.9	62.5
Liao *et al.* [27]	**99.8**	**98.8**	91.9	98.8	86.4	-	-	-	91.5	-	-	79.9
Shi *et al.* [46]	99.6	**98.8**	93.4	**97.4**	**89.5**	98.8	98.0	94.5	91.8	76.1	78.5	79.5
ScRN **(ours)**	99.5	**98.8**	**94.4**	97.2	88.9	99.0	98.3	**95.0**	93.9	**78.7**	**80.8**	**87.5**

Table 4: Results across a number of methods and datasets. "50", "1k", "Full" are lexicons. "0" means no lexicon.

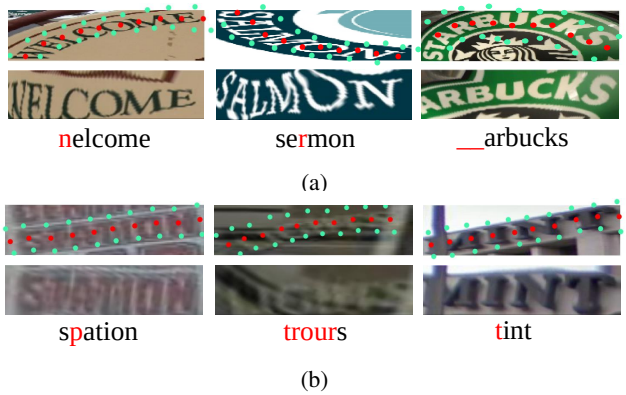

Figure 8: Some bad cases produced by our recognition system. The meanings of these elements are the same as Fig. 6. Incorrectly recognized characters are in red.

ognized, due to imperfect rectification. We observe that our rectification module suffers from the curved text whose terminal characters have a nearly horizontal orientation and are close to the image borders. In Fig. 8b, ScRN is able to give satisfactory rectification results, yet the recognizer fails to handle such blurry or occlusive cases.

Although character-level annotations are needed in our rectification module, it is labor-free and time-efficient to obtain such annotations with the automatic synthesizing engine [15]. In addition, extra images with only word-level annotations, such as Synth90k, can also be added for training to further improve the performance.

5. Conclusion

In this paper, we have proposed a Symmetry-constrained Rectification Network (ScRN) for scene text recognition. Such a flexible module can be either easily incorporated into existing recognition models or trained in an end-to-end manner within a unified framework. Our text recognition system incorporating the proposed ScRN achieves state-of-the-art performance on a number of benchmark datasets, especially on those with a large portion of irregular text images. Due to the shared backbone, ScRN significantly improves the recognition performance while requires negligible extra computation. Comprehensive experiments demonstrate the effectiveness and robustness of our recognition system. As for future work, we would like to extend the proposed method to an end-to-end text recognition system which can deal with text instances of arbitrary shapes.

Acknowledgments

This work was supported by NSFC 61733007, to Dr. Xiang Bai by the National Program for Support of Top-notch Young Professionals and the Program for HUST Academic Frontier Youth Team 2017QYTD08. In addition, we sincerely thank Shangbang Long for his help.

References

[1] J. Almazán, A. Gordo, A. Fornés, and E. Valveny. Word spotting and recognition with embedded attributes. *TPAMI*, 36(12):2552–2566, 2014. 8

[2] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014. 4

[3] F. Bai, Z. Cheng, Y. Niu, S. Pu, and S. Zhou. Edit probability for scene text recognition. In *CVPR*, 2018. 2, 8

[4] X. Bai, C. Yao, and W. Liu. Strokelets: A learned multi-scale mid-level representation for scene text recognition. *IEEE Transactions on Image Processing*, 25(6):2789–2802, 2016. 2

[5] A. Bissacco, M. Cummins, Y. Netzer, and H. Neven. Photoocr: Reading text in uncontrolled conditions. In *ICCV*, pages 785–792, 2013. 8

[6] F. L. Bookstein. Principal warps: Thin-plate splines and the decomposition of deformations. *TPAMI*, 11(6):567–585, 1989. 3

[7] Z. Cheng, F. Bai, Y. Xu, G. Zheng, S. Pu, and S. Zhou. Focusing attention: Towards accurate text recognition in natural images. In *ICCV*, pages 5086–5094, 2017. 2, 8

[8] Z. Cheng, Y. Xu, F. Bai, Y. Niu, S. Pu, and S. Zhou. Aon: Towards arbitrarily-oriented text recognition. In *CVPR*, pages 5571–5579, 2018. 1, 2, 7, 8

[9] K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1724–1734, 2014. 4

[10] B. Epshtein, E. Ofek, and Y. Wexler. Detecting text in natural scenes with stroke width transform. In *CVPR*, pages 2963–2970, 2010. 2

[11] R. Girshick. Fast r-cnn. In *ICCV*, December 2015. 5

[12] A. Gordo. Supervised mid-level features for word image representation. In *CVPR*, pages 2956–2964, 2015. 8

[13] A. Graves, S. Fernández, F. J. Gomez, and J. Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *ICML*, pages 369–376, 2006. 2

[14] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber. A novel connectionist system for unconstrained handwriting recognition. *TPAMI*, 31(5):855–868, 2009. 4

[15] A. Gupta, A. Vedaldi, and A. Zisserman. Synthetic data for text localisation in natural images. In *CVPR*, pages 2315–2324, 2016. 5, 8

[16] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 2

[17] P. He, W. Huang, Y. Qiao, C. C. Loy, and X. Tang. Reading scene text in deep convolutional sequences. In *AAAI*, volume 16, pages 3501–3508, 2016. 2

[18] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman. Synthetic data and artificial neural networks for natural scene text recognition. *CoRR*, abs/1406.2227, 2014. 5

[19] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman. Deep structured output learning for unconstrained text recognition. In *ICLR*, 2015. 8

[20] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman. Reading text in the wild with convolutional neural networks. *IJCV*, 116(1):1–20, 2016. 2, 8

[21] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In *NIPS*, pages 2017–2025, 2015. 1, 2

[22] M. Jaderberg, A. Vedaldi, and A. Zisserman. Deep features for text spotting. In *ECCV*, pages 512–528, 2014. 8

[23] D. Karatzas, L. Gomez-Bigorda, A. Nicolaou, S. K. Ghosh, A. D. Bagdanov, M. Iwamura, J. Matas, L. Neumann, V. R. Chandrasekhar, S. Lu, F. Shafait, S. Uchida, and E. Valveny. ICDAR 2015 competition on robust reading. In *Proc. ICDAR*, pages 1156–1160, 2015. 5

[24] D. Karatzas, F. Shafait, S. Uchida, M. Iwamura, L. G. i Bigorda, S. R. Mestre, J. Mas, D. F. Mota, J. A. Almazan, and L. P. de las Heras. Icdar 2013 robust reading competition. In *ICDAR*, pages 1484–1493, 2013. 5

[25] C.-Y. Lee and S. Osindero. Recursive recurrent nets with attention modeling for ocr in the wild. In *CVPR*, pages 2231–2239, 2016. 2, 8

[26] G. Li, S. Xu, X. Liu, L. Li, and C. Wang. Jersey number recognition with semi-supervised spatial transformer network. In *CVPR Workshops*, pages 1783–1790, 2018. 2, 7

[27] M. Liao, J. Zhang, Z. Wan, F. Xie, J. Liang, P. Lyu, C. Yao, and X. Bai. Scene text recognition from two-dimensional perspective. In *AAAI*, 2019. 8

[28] T. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 936–944, 2017. 2

[29] W. Liu, C. Chen, and K. K. Wong. Char-net: A character-aware neural network for distorted scene text recognition. In *AAAI*, pages 7154–7161, 2018. 2, 8

[30] Y. Liu, Z. Wang, H. Jin, and I. J. Wassell. Synthetically supervised feature learning for scene text recognition. In *ECCV*, pages 449–465, 2018. 2, 8

[31] Z. Liu, Y. Li, F. Ren, W. L. Goh, and H. Yu. Squeezedtext: A real-time scene text recognition by binary convolutional encoder-decoder network. In *AAAI*, pages 7194–7201, 2018. 8

[32] S. Long, X. He, and C. Yao. Scene text detection and recognition: The deep learning era. *arXiv preprint arXiv:1811.04256*, 2018. 1, 2

[33] S. Long, J. Ruan, W. Zhang, X. He, W. Wu, and C. Yao. Textsnake: A flexible representation for detecting text of arbitrary shapes. In *ECCV*, pages 19–35. Springer, 2018. 3

[34] S. M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, and R. Young. Icdar 2003 robust reading competitions. In *ICDAR*, page 682. IEEE, 2003. 5

[35] P. Lyu, M. Liao, C. Yao, W. Wu, and X. Bai. Mask textspotter: An end-to-end trainable neural network for spotting text with arbitrary shapes. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 67–83, 2018. 1

[36] P. Lyu, C. Yao, W. Wu, S. Yan, and X. Bai. Multi-oriented scene text detection via corner localization and region seg-

mentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7553–7563, 2018. 1

[37] A. Mishra, K. Alahari, and C. Jawahar. Scene text recognition using higher order language priors. In *BMVC*. BMVA, 2012. 2, 5

[38] A. Mishra, K. Alahari, and C. Jawahar. Top-down and bottom-up cues for scene text recognition. In *CVPR*. IEEE, 2012. 2, 8

[39] T. Novikova, O. Barinova, P. Kohli, and V. S. Lempitsky. Large-lexicon attribute-consistent text recognition in natural images. In *ECCV*, pages 752–765, 2012. 2

[40] T. Quy Phan, P. Shivakumara, S. Tian, and C. Lim Tan. Recognizing text with perspective distortion in natural scenes. In *ICCV*, pages 569–576, 2013. 2, 5

[41] A. Risnumawan, P. Shivakumara, C. S. Chan, and C. L. Tan. A robust arbitrary text detection system for natural scene images. *Expert Syst. Appl.*, 41(18):8027–8048, 2014. 5

[42] J. A. Rodríguez-Serrano, A. Gordo, and F. Perronnin. Label embedding: A frugal baseline for text recognition. *IJCV*, 113(3):193–207, 2015. 8

[43] X. Rong, C. Yi, and Y. Tian. Recognizing text-based traffic guide panels with cascaded localization network. In *ECCV Workshops*, pages 109–121, 2016. 1

[44] B. Shi, X. Bai, and C. Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *TPAMI*, 39(11):2298–2304, 2017. 2, 8

[45] B. Shi, X. Wang, P. Lyu, C. Yao, and X. Bai. Robust scene text recognition with automatic rectification. In *CVPR*, pages 4168–4176, 2016. 1, 2, 8

[46] B. Shi, M. Yang, X. Wang, P. Lyu, C. Yao, and X. Bai. Aster: an attentional scene text recognizer with flexible rectification. *TPAMI*, 2018. 1, 2, 8

[47] B. Su and S. Lu. Accurate scene text recognition based on recurrent neural network. In *ACCV*, pages 35–48, 2014. 8

[48] B. Su and S. Lu. Accurate recognition of words in scenes without character segmentation using recurrent neural network. *Pattern Recognition*, 63:397–405, 2017. 2

[49] K. Wang, B. Babenko, and S. J. Belongie. End-to-end scene text recognition. In *ICCV*, pages 1457–1464, 2011. 2, 5, 8

[50] K. Wang and S. Belongie. Word spotting in the wild. In *ECCV*, pages 591–604, 2010. 2

[51] T. Wang, D. J. Wu, A. Coates, and A. Y. Ng. End-to-end text recognition with convolutional neural networks. In *ICPR*, pages 3304–3308, 2012. 2, 8

[52] X. Yang, D. He, Z. Zhou, D. Kifer, and C. L. Giles. Learning to read irregular text with attention mechanisms. In *IJCAI*, pages 3280–3286, 2017. 2, 8

[53] C. Yao, X. Bai, and W. Liu. A unified framework for multi-oriented text detection and recognition. *IEEE Transactions on Image Processing*, 23(11):4737–4749, 2014. 1

[54] C. Yao, X. Bai, W. Liu, Y. Ma, and Z. Tu. Detecting texts of arbitrary orientations in natural images. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1083–1090. IEEE, 2012. 1

[55] C. Yao, X. Bai, B. Shi, and W. Liu. Strokelets: A learned multi-scale representation for scene text recognition. In *CVPR*, pages 4042–4049, 2014. 2, 8

[56] M. D. Zeiler. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701, 2012. 6

[57] F. Zhan, S. Lu, and C. Xue. Verisimilar image synthesis for accurate detection and recognition of texts in scenes. In *ECCV*, pages 257–273. Springer, 2018. 2

[58] Z. Zhang, W. Shen, C. Yao, and X. Bai. Symmetry-based text line detection in natural scenes. In *CVPR*, pages 2558–2567, 2015. 3

[59] X. Zhou, C. Yao, H. Wen, Y. Wang, S. Zhou, W. He, and J. Liang. East: an efficient and accurate scene text detector. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 5551–5560, 2017. 1

[60] Y. Zhu, C. Yao, and X. Bai. Scene text detection and recognition: Recent advances and future trends. *Frontiers of Computer Science*, 10(1):19–36, 2016. 1