ABCNet v2: Adaptive Bezier-Curve Network for Real-Time End-to-End Text Spotting

Yuliang Liu, Chunhua Shen, Lianwen Jin, *Member, IEEE*, Tong He, Peng Chen, Chongyu Liu, and Hao Chen

Abstract—End-to-end text-spotting, which aims to integrate detection and recognition in a unified framework, has attracted increasing attention due to its simplicity of the two complimentary tasks. It remains an open problem especially when processing arbitrarily-shaped text instances. Previous methods can be roughly categorized into two groups: character-based and segmentation-based, which often require character-level annotations and/or complex post-processing due to the unstructured output. Here, we tackle end-to-end text spotting by presenting Adaptive Bezier Curve Network v2 (ABCNet v2). Our main contributions are four-fold: 1) For the first time, we adaptively fit arbitrarily-shaped text by a parameterized Bezier curve, which, compared with segmentation-based methods, can not only provide structured output but also controllable representation. 2) We design a novel BezierAlign layer for extracting accurate convolution features of a text instance of arbitrary shapes, significantly improving the precision of recognition over previous methods. 3) Different from previous methods, which often suffer from complex post-processing and sensitive hyper-parameters, our ABCNet v2 maintains a simple pipeline with the only post-processing non-maximum suppression (NMS). 4) As the performance of text recognition closely depends on feature alignment, ABCNet v2 further adopts a simple yet effective coordinate convolution to encode the position of the convolutional filters, which leads to a considerable improvement with negligible computation overhead. Comprehensive experiments conducted on various bilingual (English and Chinese) benchmark datasets demonstrate that ABCNet v2 can achieve state-of-the-art performance while maintaining very high efficiency. More importantly, as there is little work on quantization of text spotting models, we quantize our models to improve the inference time of the proposed ABCNet v2. This can be valuable for real-time applications. Code and model are available at: https://git.io/AdelaiDet

Index Terms—Bezier curve, scene text spotting, text detection and recognition

1 Introduction

Text spotting in the natural environment has drawn increasing research attention in the community of computer vision and image understanding, which aims to detect and recognize text instances in unconstrained conditions. Text information recovered has proven valuable for image retrieval, automatic organization of photos and visual assistance, to name a few. To date, it remains challenging for the following reasons: 1) text instances often exhibit diverse patterns in shape, color, font, and language. These inevitable variations in the data often require heuristic settings for achieving satisfactory performance; 2) real-time applications require the algorithm to achieve a better trade-off between the efficiency and effectiveness. Although the emergence of deep learning has significantly improved the performance of the task of scene text spotting, current methods still exist a considerable gap for solving generic real-world applications.

Previous approaches often involve two independent modules for scene text spotting: text detection and recognition, which are implemented sequentially. Many of them only address one task and directly borrow top-performing modules from the other. Such a simplified approach is unlikely to exploit the full potential of deep convolution networks, as two tasks are isolated without shareable features.

Recently, end-to-end scene text spotting methods [3], [4], [5], [6], [7], [8], [9], [10], [11], which directly build the mapping between the input image and sets of words transcripts in a unified framework, is drawing increasing attention. Compared to the models in which the detection and recognition are two separate modules, the advantages of designing an end-to-end framework are as follows. First, word recognition can significantly improve the accuracy of detection. One of the most notable characteristics for text is the attribute of being sequences. However, false positives exhibiting the appearance of sequences exist in unconditioned environments, such as blocks, buildings, and railings. To empower the network to have discriminative capability to distinguish different patterns, some approaches [3], [4], [5] propose to share the features between the two tasks and train the network in an end-to-end manner. Moreover, due to the shared features, end-to-end frameworks often show superiority in the inference speed, which is more suitable for real-time

- *Yuliang Liu is with the South China University of Technology, Huizhou 516057, China, and also with the University of Adelaide, Adelaide, SA 5005, Australia. E-mail: liu.yuliang@mail.scut.edu.cn.*
- *Chunhua Shen, Tong He, Peng Chen, and Hao Chen are with the University of Adelaide, Adelaide, SA 5005, Australia. E-mail: {chhshen, stanzju}@gmail.com, {tong.he, peng.chen}@adelaide.edu.au.*
- *Lianwen Jin is with the South China University of Technology, Huizhou 516057, China, and also with the Guangdong Artificial Intelligence and Digital Economy Laboratory, Guangzhou, Guangdong 510335, China. E-mail: eelwjin@scut.edu.cn.*
- *Chongyu Liu is with the South China University of Technology, Huizhou 516057, China. E-mail: liuchongyu1996@gmail.com.*

Manuscript received 13 Dec. 2020; revised 19 July 2021; accepted 21 Aug. 2021. Date of publication 30 Aug. 2021; date of current version 3 Oct. 2022. (Corresponding authors: Chunhua Shen and Lianwen Jin.) Recommended for acceptance by X. Bai. Digital Object Identifier no. 10.1109/TPAMI.2021.3107437

(a) TPS-based alignment. (b) BezierAlign-ed results.

Fig. 1. *Comparison of the warping results*. In Figure (a), we follow previous methods by using TPS [1] and STN [2] to warp the curved text region into a rectangular shape. In Figure (b), we use generated Bezier curves and the proposed BezierAlign to warp the results, leading to improved accuracy.

applications. Finally, current standalone recognition models usually adopt perfectly cropped text images or heuristic synthetic images for training. An end-to-end module can force the recognition module to accustom to the detection outputs, and thus the results can be more robust [10].

Existing approaches for end-to-end text-spotting can be roughly categorized into two groups: character-based and segmentation-based. Character-based methods first detect and recognize individual characters and then output words by applying an extra grouping module. Although effective, laborious character-level annotations are required. Besides, often a few predefined hyper-parameters are necessary for the grouping algorithm, showing limited robustness and generalization capability. Another line of the research is segmentation based, where text instances are represented by unstructured contours, making it difficult for the subsequent recognition step. For example, the work in [12] relies on an TPS [1] or an STN [2] step to warp the original ground truths into rectangular shape. Note that, the characters can be significantly distorted, as shown in Fig. 1. Besides, compared with detection, text recognition requires a significantly large amount of training data, resulting in optimization difficulties in a unified framework.

To address these limitations, we propose the Adaptive Bezier Curve Network v2 (ABCNet v2), an end-to-end trainable framework, for real-time arbitrarily shaped scene text spotting. ABCNet v2 enables arbitrarily shaped scene text detection with simple yet effective Bezier curve adaptation, which introduces negligible computation overhead compared with standard rectangle bounding box detection. In addition, we design a novel feature alignment layer, termed BezierAlign, to precisely calculate convolutional features of text instances in curved shapes, and thus high recognition accuracy can be achieved. *For the first time, we successfully adopt the parameter space (Bezier curves) for multi-oriented or curved text spotting, enabling a very concise and efficient pipeline.*

Inspired by recent work in [13], [14], [15], we improve ABCNet in our conference version [16] in four aspects: the feature extractor, detection branch, recognition branch, and end-to-end training. Due to the inevitable variation of scales, the proposed ABCNet v2 incorporates iterative bidirectional features to achieve a better accuracy and efficiency trade-off. In addition, based on our observations, feature alignment in the detection branch is essential for the subsequent text recognition. To this end, we adopt a coordinate encoding approach with negligible computation overhead to explicitly encode the position in the convolutional filters, leading to considerable improvement in accuracy. For the recognition branch, we integrate a character attention module, which can recursively predict the characters of each word without using character-level annotations. To enable effective end-to-end training, we further propose an Adaptive End-to-End Training (AET) strategy to match detection for end-to-end training. This can force the recognition branch more robust to the detection behaviors.

Thus, the proposed ABCNet v2 enjoys several advantages over previous state-of-the-art methods, which are summarized as follows:

- For the first time, we introduce a new, concise parametric representation of curved scene text using Bezier curves. It introduces negligible computation overhead compared with the standard bounding box representation.
- We propose a new feature alignment method, *a.k.a.* BezierAlign, and thus the recognition branch can be seamlessly connected to the overall structure. By sharing backbone features, the recognition branch can be designed with a light-weight structure for efficient inference.
- The detection model of ABCNet v2 is more general for processing multi-scale text instances by considering bidirectional multi-scale pyramid global textual features.
- To our knowledge, our method is the first framework that can simultaneously detect and recognize horizontal, multi-oriented, and arbitrarily shaped text in a single-shot manner, while keeping a real-time inference speed.
- To further speed up inference, we also exploit the technique of model quantization, showing that ABCNet v2 can reach a much faster inference speed with only marginal accuracy reduction.
- Comprehensive experiments on various benchmarks demonstrate the state-of-the-art text spotting performance of the proposed ABCNet v2 in terms of accuracy and speed.

2 Related Work

Scene text spotting requires detecting and recognizing text simultaneously, instead of concerning only one task. In the early days, scene text spotting methods are usually simply connected by the independent detection and recognition models. Two models are separately optimized with different architectures. Recently, end-to-end methods (Section 2.2) have significantly advanced the performance of text spotting by integrating the detection and recognition into one unified network.

2.1 Separate Scene Text Spotting

In this section, we briefly review the literature, focusing on either detection or recognition.

2.1.1 Scene Text Detection

The development trend of text detection can be observed through the detection flexibility. From focused horizontal

scene text detection represented by horizontal rectangular detection bounding boxes, to multi-oriented scene text detection represented by rotated rectangular or quadrilateral bounding boxes, and to arbitrarily shaped scene text detection represented by instance segmentation masks or polygons.

The early horizontal rectangular based methods can be dated back to Lucas *et al.* [17], in which the pioneering horizontal ICDAR'03 benchmark is constructed. ICDAR'03 and its successive datasets (ICDAR'11 [18] and ICDAR'13 [19]) have attracted considerable research efforts [20], [21], [22], [23], [24], [25] in studies on horizontal scene text detection.

Before 2010, most of the methods merely focus on regular horizontal scene text, which is limited to generalize to the real applications, where multi-oriented scene text is ubiquitous. To this end, Yao *et al.* [26] put forward a practical detection system as well as a multi-oriented benchmark (MSRA-TD500) for multi-oriented scene text detection. Both the method and the dataset use rotated rectangular bounding boxes to detect and annotate the multi-oriented text instances. Besides MSRA-TD500, the emergence of other multi-oriented datasets including NEOCR [27], and USTB-SV1K [28] further facilitate numerous rotated rectangle based methods [3], [26], [28], [29], [30], [31]. Since 2015, ICDAR'15 [32] starts to use four points based quadrilateral annotations for each text instance, which facilitates numerous methods that successfully demonstrate the superiority of the tighter and more flexible quadrilateral detection methods. The SegLink method [33] predicts text regions by oriented segments and learns connecting links to recombine the results. DMPNet [34] observes that rotated rectangles may still contain unnecessary background noises, imperfect matching, or unnecessary overlap, and thus it proposes the use of quadrilateral bounding boxes to detect text with auxiliary, predefined quadrilateral sliding windows. EAST [35] employs a dense prediction structure for directly predicting quadrilateral bounding boxes. WordSup [36] proposes an iterative strategy to generate characters region automatically, which shows robustness on complicated scenes. The successful attempt of the ICDAR 2015 motivates numerous quadrilateral based datasets, such as RCTW'17 [37], MLT [38], and ReCTS [39].

Recently, the research focus has shifted from multi-oriented scene text detection to arbitrarily shaped text detection. The arbitrary shape is mainly presented by curved text in the wild, which can also be very common in our real world, e.g., columnar objects (bottles and stone piles), spherical objects, plicate planes (clothes, streamer, and receipts), coins, logos, seal, signboard and so on. The first curved text dataset CUTE80 [40] is constructed in 2014. But this dataset is mainly used for scene text recognition, as it contains only 80 clean images with relative few text instances. For detection on arbitrarily shaped scene text, two recent benchmarks—Total-Text [41] and SCUT-CTW1500 [42]—have been proposed to advance many influential works [43], [44], [45], [46], [47], [48], [49], [50], [51]. TextSnake [47] designs an FCN to predict the geometry attributes of text instances and then groups them into the final output. CRAFT [44] predicts the character regions of the text and the affinity between the adjacent ones. SegLink++ [48] proffers an instance-aware component grouping framework for dense and arbitrary shaped text detection. PSENet [46] proposes to learn the text kernel, then expands them to cover the whole text instances. PAN [45] based on PSENet [46], adopts a learnable post-processing method by predicting similarity vectors of pixels. Wang *et al.* [52] propose to learn the adaptive text region representation for the detection of text in arbitrary shape. DRRN [53] proposes to first detect text components, and then groups them together through a graph network. ContourNet [50] adopts an adaptive-RPN and an extra contour prediction branch to improve the precision.

2.1.2 Scene Text Recognition

Scene text recognition aims at recognizing text through a cropped text image. Many previous methods [54], [55], following the bottom-up approach, first segment character regions through sliding windows and classify each character, then group them into a word for taking the dependence with its neighbors into consideration. They achieve good performance in scene text recognition, but are limited to costly character-level annotations for character detection. Without large training datasets, the models' in this category typically cannot generalize well. The works proposed by Su and Lu [56], [57] present a scene text recognition system by using HOG feature and Recurrent Neural Network (RNN), which are one of pioneer works that successfully introduce RNN for scene text recognition. Later, CNN-based methods with recurrent neural network are proposed to perform in a top-down manner, which can end-to-end predict a text sequence without any character detection. Shi *et al.* [58] apply Connectionist Temporal Classification (CTC) [59] upon a network integrated CNNs with RNNs, termed CRNN. Guided by the CTC loss, CRNN-based model can effectively transcribe the image content. Besides CTC, the attention mechanism [60] is also employed for text recognition.

The above methods are mainly applied to regular text recognition and not sufficiently robust for irregular one. In recent years, approaches for arbitrary-shape text recognition become dominant, which can be categorized into rectification-based methods and rectification-free methods. For the former, STN [2] and Thin-Plate-Spline (TPS) [61] are two widely used methods for text rectification. Shi *et al.* [62] are the first to introduce STN and the attention-based decoder for predicting the text sequence. The work in [63] achieves better performance using iterative text rectification. Besides, Luo *et al.* [64] propose MORAN, which rectifies the text through regressing the offsets for location shift. Liu *et al.* [65] propose a Character-Aware Neural Network (CharNet), which detects characters first and then separately transforms them into horizontal one. ESIR [66] presents an iterative rectification pipeline that can turn the position of text from perspective distortion to regular format, and thus an effective end-to-end scene text recognition system can be built. Litman *et al.* [67] first apply TPS on input images and then stack several selective attention decoder for both visual and contextual features.

In the category of rectification-free methods, Cheng *et al.* [68] propose an arbitrary orientation network (AON) to extract features in four directions and the character position clues. Li *et al.* [69] apply the 2D-attention mechanism to capture irregular text features and achieved impressive results.

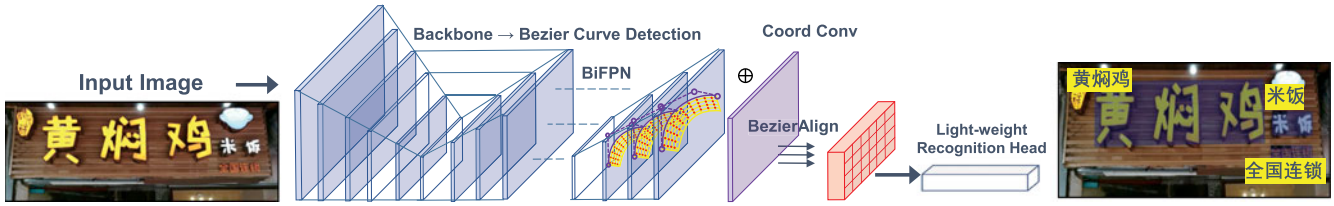


Fig. 2. *The framework of the proposed ABCNet v2*. We use cubic Bezier curves and BezierAlign to extract multi-scale curved sequence features using the Bezier curve detection results. We concatenate coordinate channels to encode the position coordinates in FPN output features before sending to BezierAlign. The overall framework is end-to-end trainable with high efficiency. Here purple dots represent the control points of the cubic Bezier curve.

To tackle the attention drift issue, Yue *et al.* [70] design a novel position enhancement branch in the recognition model. In addition, some rectification-free methods are based on semantic segmentation. Liao *et al.* [71] and Wan *et al.* [72] both propose to segment characters and classify their categories through visual features.

2.2 End-to-End Scene Text Spotting

2.2.1 Regular End-to-End Scene Text Spotting

Li *et al.* [3] may be the first to propose an end-to-end trainable scene text spotting method. The method successfully uses an RoI Pooling [73] to join detection and recognition features via a two-stage framework. It is designed to process horizontal and focused text. Its improved version [11] significantly improves the performance. Busta *et al.* [74] also propose an end-to-end deep text spotter. He *et al.* [4] and Liu *et al.* [5] adopt an anchor-free mechanism to improve both the training and inference speed. They use a similar sampling strategy, i.e., Text-Align-Sampling and RoI-Rotate, respectively, to enable extracting features from quadrilateral detection results. Note that both of these two methods are not capable of spotting arbitrarily-shaped scene text.

2.2.2 Arbitrarily-Shaped End-to-End Scene Text Spotting

To detect arbitrarily-shaped scene text, Liao *et al.* [6] propose a Mask TextSpotter which subtly refines Mask R-CNN and uses character-level supervision to simultaneously detect and recognize characters and instance masks. The method significantly improves the performance of spotting arbitrarily-shaped scene text. Its improved version [10] significantly alleviates the reliance on the character-level annotations. Sun *et al.* [75] propose the TextNet which produces quadrilateral detection bounding boxes in advance, and then use a region proposal network to feed the detection features for recognition.

Recently, Qin *et al.* [7] propose to use a RoI Masking to focus on the arbitrarily-shaped text region. Note that extra computation is needed to fit polygons. The work in [8] propose an arbitrarily-shaped scene text spotting method, termed CharNet, requiring character-level training data and TextField [76] to group recognition results. Authors of [9] propose a novel sampling method, RoISlide, which uses fused features from the predicting segments of the text instances, and thus it is robust to long arbitrarily-shaped text.

Wang *et al.* [12] first detect the boundary points of text in arbitrary shapes, through TPS to rectify the detected text and then fed it into the recognition branch. Liao *et al.* [77] propose a Segmentation Proposal Network (SPN) to accurate extract the text regions, and it follows [10] to attain the final results.

3 Our Method

An intuitive pipeline of our method is shown in Fig. 2. Inspired by [78], [79], [80], we adopt a single-shot, anchor-free convolutional neural network as the detection framework. Removal of anchor boxes significantly simplifies the detection for our task. Here the detection is densely predicted on the output feature maps of the detection head, which is constructed by 4 stacked convolution layers with stride of 1, padding of 1, and 3×3 kernels. Next, we present the key components of the proposed ABCNet v2 in six components: 1) Bezier curve detection; 2) the coordinate convolution module; 3) BezierAlign; 4) the light-weight attention recognition module; 5) the adaptive end-to-end training strategy; and 6) text spotting quantization.

3.1 Bezier Curve Detection

Compared to segmentation-based methods [44], [46], [47], [49], [76], [81], regression-based methods are more suitable for arbitrarily-shaped text detection, as demonstrated in [42], [52]. A drawback of these methods is the complicated pipeline, and they often require complex post-processing steps to obtain the final results.

To simplify the detection for arbitrarily-shaped scene text instances, we propose to fit a Bezier curve by regressing several key points. The Bezier curve represents a parametric curve $c(t)$ that uses the Bernstein Polynomials as its basis. The definition is shown in Equation (1):

$$c(t) = \sum_{i=0}^{n} b_i B_{i,n}(t), \quad 0 \le t \le 1, \tag{1}$$

where, n represents the degree, b_i represents the i-th control points, and $B_{i,n}(t)$ represents the Bernstein basis polynomials, as shown in Equation (2):

$$B_{i,n}(t) = \binom{n}{i} t^i (1-t)^{n-i}, \quad i = 0, \ldots, n, \tag{2}$$

where $\binom{n}{i}$ is the binomial coefficient. To fit arbitrary shapes of the text with Bezier curves, we examine the arbitrarily-shaped scene text from the existing datasets and we empirically show that a cubic Bezier curve (i.e., $n = 3$) is sufficient to fit different formats of curved scene text, especially on dataset with word-level annotation. A higher-order may work better on text-line level datasets, where multiple waves may be presented in one instance. We provide comparisons in terms of the order of the Bezier curves in the

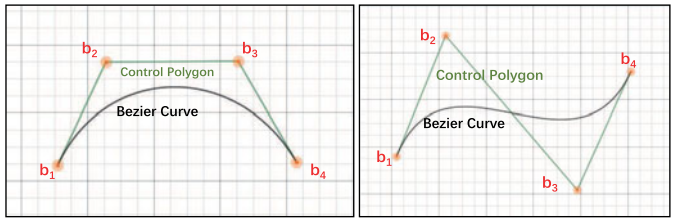


Fig. 3. *Cubic Bezier curves.* b_i represents the control points. The green lines form a control polygon, and the black curve is the cubic Bezier curve. Note that with only two end-points b_1 and b_4, the Bezier curve degenerates to a straight line.

experiment section. An illustration of cubic Bezier curves is shown in Fig. 3.

Based on the cubic Bezier curve, we can formulate the arbitrarily-shaped scene text detection into a regression problem similar to bounding box regression, but with eight control points in total. Note that a straight text that has four control points (four vertexes) is a typical case of arbitrarily-shaped scene text. For consistency, we interpolate additional two control points in the tripartite points of each long side.

To learn the coordinates of the control points, we first generate the Bezier curve annotations described in Section 3.1.1 and follow a similar regression method as in [34] to regress the targets. For each text instance, we use

$$\Delta_x = b_{ix} - x_{min}, \quad \Delta_y = b_{iy} - y_{min}, \tag{3}$$

where x_{min} and y_{min} represent the minimum x and y values of the 4 vertexes, respectively. The advantage of predicting the relative distance is that it is irrelevant to whether the Bezier curve control points are beyond the image boundary. Inside the detection head, we only use one convolution layer with $4(n+1)$ (n is the number of Bezier Curve order) output channels to learn the Δ_x and Δ_y, which is nearly cost-free while the results can still be accurate. We discuss the details in Section 4.

3.1.1 Bezier Ground-Truth Generation

In this section, we briefly introduce how to generate Bezier curve ground-truth based on the original annotations. The arbitrarily-shaped datasets, e.g., Total-text [41] and SCUT-CTW1500 [42], use polygonal annotations for the text regions. Given the annotated points $\{p_i\}_{i=1}^{n}$ from the curved boundary, where p_i represents the i-th annotating point, the main goal is to obtain the optimal parameters for cubic Bezier curves $c(t)$ in Equation (1). To achieve this, we can simply apply the standard least-squares fitting as follows:

$$\begin{bmatrix} B_{0,n}(t_0) & \dots & B_{n,n}(t_0) \\ B_{0,n}(t_1) & \dots & B_{n,n}(t_1) \\ \vdots & \ddots & \vdots \\ B_{0,n}(t_m) & \dots & B_{n,n}(t_m) \end{bmatrix} \begin{bmatrix} b_{x_0} & b_{y_0} \\ b_{x_1} & b_{y_1} \\ \vdots & \vdots \\ b_{x_n} & b_{y_n} \end{bmatrix} = \begin{bmatrix} p_{x_0} & p_{y_0} \\ p_{x_1} & p_{y_1} \\ \vdots & \vdots \\ p_{x_m} & p_{y_m} \end{bmatrix}. \tag{4}$$

Here m represents the number of the annotated points for a curved boundary. For Total-Text and SCUT-CTW1500, m is 5 and 7, respectively. t is calculated by using the ratio of the cumulative length to the perimeter of the poly-line. According to Equations (1) and (4), we convert the original poly-line annotation to a parameterized Bezier curve. Note that we directly use the first and the last annotating points as the first (b_0) and the last (b_n) control points, respectively. An visualization comparison is shown in Fig. 1, which shows that the generated results can be even visually better than the original annotation. In addition, thanks to the structured output, the task of text recognition can be easily formulated by applying our proposed BezierAlign (see Section 3.3), which warps the curved text into horizontal representation. More results of the Bezier curve generation are shown in Fig. 4. The simplicity of our method allows it to deal with various shapes in a unified representation format.

3.2 CoordConv

As pointed out in [14], conventional convolutions show limitation when learning a mapping between coordinates in (x, y) Cartesian space and coordinates in one-hot pixel space. The problem can be effectively solved by concatenating the coordinates to the feature maps. The recent practice of encoding relative coordinates [15] also show that the relative coordinates can provide informative cues for instance segmentation.

Let f_{outs} denotes the features of different scales of FPN, and $O_{i,x}$ and $O_{i,y}$ represent the absolute x and y coordinates, respectively, from all the locations (i.e., the location where the filters are generated) for the ith level of FPN. All $O_{i,x}$ and $O_{i,y}$ consist of two feature maps f_{ox} and f_{oy}. We simply concatenate f_{ox} and f_{oy} to the last channel of f_{outs} along the channel dimension. Therefore, new features f_{coord} with additional two channels are formed, which are subsequently input to three convolutional layers with kernel size, stride, and padding size setting to 3, 1, and 1, respectively. We find that using such simple coordinate convolutions can considerably improve the performance of scene text spotting.

3.3 BezierAlign

To enable end-to-end training, most of the previous methods adopt various sampling (feature alignment) methods to connect the recognition branch. Typically, a sampling method represents an in-network region cropping procedure. In other words, given a feature map and Region-of-Interest (RoI), using the sampling method to extract the features of RoI and efficiently output a feature map of a fixed size. However, sampling methods of previous non-segmentation based methods, e.g., RoI Pooling [3], RoI-Rotate [5], Text-Align-Sampling [4], or RoI Transform [75] cannot properly align features of arbitrarily-shaped text. By exploiting the parameterization nature of a structured Bezier curve bounding box, we propose BezierAlign for feature sampling/alignment, which may be viewed as a flexible extension of RoIAlign [82]. Unlike RoIAlign, the shape of the sampling grid of BezierAlign is not rectangular. Instead, each column of the arbitrarily-shaped grid is orthogonal to the Bezier curve boundary of the text. The sampling points have equidistant interval in width and height, respectively, which are bilinear interpolated with respect to the coordinates.

Formally, given an input feature map and Bezier curve control points, we process all the output pixels of the rectangular output feature map with size $h_{out} \times w_{out}$. Taking pixel

Fig. 4. *Example results of Bezier curve generation.* Green lines are the final Bezier curve results. Red dash lines represent the control polygon, and the 4 red end points represent the control points. Zoom in for better visualization.

(a) Horizontal sampling. (b) Quadrilateral sampling. (c) BezierAlign.

Fig. 5. *Comparison between previous sampling methods and BezierAlign.* The proposed BezierAlign can more accurately sample features of the text region, which is essential for achieving good recognition accuracy. Note that the alignment procedure is applied to intermediate convolution features.

g_i from output feature map with position (g_{iw}, g_{ih}) as an example, we calculate t as follows:

$$t = \frac{g_{iw}}{w_{out}}. \tag{5}$$

Then the points of upper Bezier curve boundary tp and lower Bezier curve boundary bp are calculated according to Equation (1). Using tp and bp, we can linearly index the sampling point op by Equation (6):

$$op = bp \cdot \frac{g_{ih}}{h_{out}} + tp \cdot \left(1 - \frac{g_{ih}}{h_{out}}\right). \tag{6}$$

With the position of op, we can easily apply bilinear interpolation to calculate the result. Due to the accurate sampling of features, the performance of text recognition is improved substantially. We compare BezierAlign with other sampling strategies, as presented in Fig. 5.

3.4 Attention-Based Recognition Branch

Benefiting from the shared backbone features and BezierAlign, we design a light-weight recognition branch as shown in Table 1, for faster execution. It consists of 6 convolutional layers, 1 bidirectional LSTM layer, and an attention-based recognition module. In the conference version [16], we have applied the CTC loss [59] for text string alignment between predictions and ground-truth, but we find that the attention-based recognition module [10], [60], [83], [84] is more powerful and can lead to better results. In the inference phase, the RoI region is replaced by the detected Bezier curve as in Section 3.1. Note that in [16], we only use the generated Bezier curves to extract the RoI features during training. In this paper, we also take advantages of the detection results (see Section 3.5).

The attention mechanism takes zero RNN initial states and the embedding features of an initial symbol for the sequential prediction. During each step, the c-category softmax prediction (representing the predicted character), previous hidden state, and a weighted sum of the cropped Bezier curved features are recursively used to compute the results. The prediction continues until an End-of-Sequence (EOS) symbol is predicted. The number of the class is set to 96 (excluding the EOS symbol) for English, while for the bilingual task including Chinese and English, the number of the class is set to 5462. Formally, at time step t, the attention weights are calculated by

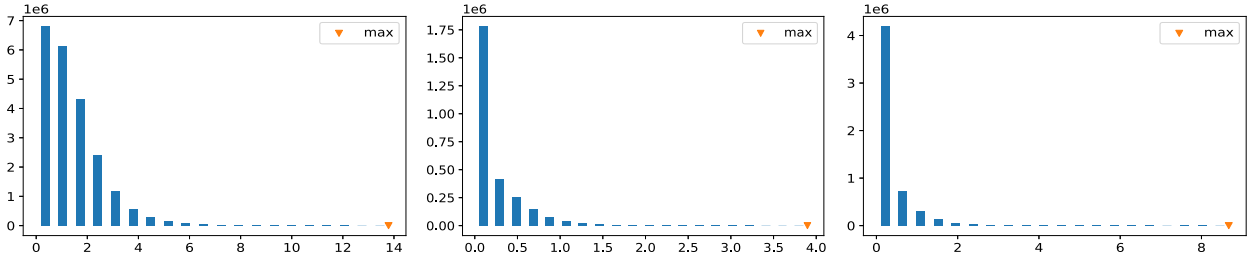
$$e_{t,s} = \boldsymbol{K}^T \tanh(\boldsymbol{W} h_{t-1} + \boldsymbol{U} h_s + \boldsymbol{b}), \tag{7}$$

where, h_{t-1} is the last hidden state, $\boldsymbol{K}$, $\boldsymbol{W}$, $\boldsymbol{U}$, and $\boldsymbol{b}$ are the learnable weight matrices and parameters. The

TABLE 1
Structure of the Recognition Branch, Which is a Simplified Version of CRNN [58]

Layers (CNN - RNN)	Parameters (kernel size, stride)	Output Size (n, c, h, w)
conv. layers $\times 4$	(3, 1)	(n, 256, h, w)
conv. layers $\times 2$	(3, (2,1))	(n, 256, h, w)
average pool for h	-	(n, 256, 1, w)
Channels-Permute	-	(w, n, 256)
BLSTM	-	(w, n, 512)
Attention-based decoder	-	(w, n, n_{class})

For all convolutional layers, the padding size is restricted to 1. n represents batch size. c represents the channel size. h and w represent the height and width of the outputted feature map, and n_{class} represents the number of the predicted class.



(a) 1st layer activation (b) 21st layer activation (c) 31st layer activation

Fig. 6. *Data distribution in the ABCNet v2 with the max value marked in the figure.* We can see that the abnormal large values have a low occurrence frequency.

weighted sum of the sequential feature vectors is formulated as

$$c_t = \sum_{s=1}^{n} a_{t,s} h_s, \tag{8}$$

where $a_{t,s}$ is defined as

$$a_{t,s} = \frac{\exp(e_{t,s})}{\sum_{s=1}^{n} \exp(e_{t,s})}. \tag{9}$$

Then, the hidden state can be updated, as follows:

$$h_t = GRU((\boldsymbol{embed}_{t-1}, c_t), h_{t-1}). \tag{10}$$

Here $\boldsymbol{embed}_{t-1}$ is an embedding vector of the previous decoding result y_t, which is generated by the classifier

$$y_t = \boldsymbol{w} h_t + \boldsymbol{b}. \tag{11}$$

Therefore, we use the softmax function to estimate the probability distribution $p(u_t)$

$$u_t = \text{softmax}(\boldsymbol{V}^T h_t), \tag{12}$$

where V represents the parameters to be learned. To stabilize training, we also use a teacher enforcing strategy [85], which delivery a ground-truth character instead of the prediction of the GRU for the next prediction under a predefined probability setting to 0.5 in our implementation.

3.5 Adaptive End-to-End Training

In our published conference version [16], we only use the ground truth to BezierAlign for the text recognition branch during the training phase. While in the testing phase, we use the detection results for feature cropping. Based on the observations, some errors may occur when the detection results are not as accurate as the ground-truth Bezier curved bounding box. To alleviate such issues, we propose a simple yet effective strategy, termed Adaptive End-to-End Training (AET).

Formally, the detection results are first suppressed by confidence thresholds, and then using NMS to eliminate the redundant detection results. The corresponding recognition ground truth is then assigned to each detection result based on the minimum summation of the distances of the coordinates of control points

$$rec = \underset{rec^* \in cp^*}{\arg\min} \sum_{i=1}^{n} |cp^*_{x_i,y_i} - cp_{x_i,y_i}|, \tag{13}$$

where cp_* is the ground truth of the control points. n is the number of the control points. After assigning the recognition annotation to the detection results, we simply concatenate the new target to the original ground truth set for further recognition training.

3.6 Text Spotting Quantization

Scene text reading applications usually require a real-time performance; however, there are few works attempting to use quantization for scene text spotting task. Model quantization aims at discretizing the full precision tensors into low-bit tensors without much degradation of network performance. Limited number of representation levels (quantization levels) are available. Given quantization bit width to be b-bit, the number of quantization levels are 2^b. It is easy to see that deep learning models might suffer significant performance drop with the quantization bit width becoming low. To maintain the accuracy, the discretization error should be minimized

$$Q^*(x) = \underset{Q}{\arg\min} \sum (Q(x) - x)^2, \tag{14}$$

where $Q(x)$ is the quantization function. Motivated by the LSQ [86], we employ the following equations as the activation quantifiers in this paper. Specifically, for any data x^a from the activation tensor X^a, its quantization value $Q(x^a)$ is computed by a serial of transformations.

First, as indicated in the work PACT [87], not all the full precision data should be linearly mapped to the quantized value. It is common to find some abnormal large values, which rarely occur in the full precision tensor. We further visualize data distribution of some layers from the ABCNet v2 in Fig. 6 and observe a similar phenomenon. Therefore, a learnable parameter $\boldsymbol{\alpha}^a$ is assigned to dynamically depict the discretization range, with beyond data clipped to the boundary

$$y^a = \min(\max(x^a, 0), \boldsymbol{\alpha}^a), \tag{15}$$

Second, data in the clipped range (so-called quantization interval) is linearly mapped to nearby integer, as shown in Equation (16)

$$z^a = \left\lfloor \frac{y^a}{\boldsymbol{\alpha}^a} \cdot (l-1) \right\rceil, \tag{16}$$

where $l = 2^b$ is the number of quantization levels mentioned above and $\lfloor \cdot \rceil$ is the nearest-rounding function. Third, to

keep the data magnitude similar before and after quantization, we apply corresponding scale factor on z^a to obtain $Q(x^a)$ by

$$Q(x^a) = z^a \cdot \frac{\alpha^a}{l-1}. \tag{17}$$

In summary, the quantization of activations can be written as

$$\begin{aligned} Q(x^a) &= \left\lfloor \min(\max(x^a, 0), \alpha^a) \cdot \frac{l-1}{\alpha^a} \right\rceil \cdot \frac{\alpha^a}{l-1} \\ &= \left\lfloor \min(\max\left(\frac{x^a}{\alpha^a}, 0\right), 1) \cdot (l-1) \right\rceil \cdot \frac{\alpha^a}{l-1}. \end{aligned} \tag{18}$$

Different from activations, weight parameters generally contain both positive and negative values, thus extra linear transforms are introduced before discretization as follows:

$$\begin{aligned} z^w &= \left\lfloor \left(\min(\max\left(\frac{x^w}{\alpha^w}, -1\right), 1) + 1 \right) / 2 \cdot (l-1) \right\rceil, \\ Q(x^w) &= \left(\frac{z^w}{l-1} \cdot 2 - 1 \right) \cdot \alpha^w = (2 \cdot z^w - (l-1)) \cdot \frac{\alpha^w}{l-1}. \end{aligned} \tag{19}$$

One issue for model quantization is the gradient vanishing caused by the round function ($\lfloor \cdot \rceil$). This is because the round function has an almost-everywhere zero gradient. Straight through estimator (STE) is employed to tackle the problem. Specifically, we override the derivative of the round function constantly to be 1 ($\partial \lfloor \cdot \rceil = 1$). We employ mini-batch gradient descent optimizer to train the quantization related parameters α^a and α^w in each layer together with the original parameters from the network.

It should be noted that for each convolutional layer, its quantization introduced parameter α^a and α^w are shared for all elements in the input activation tensor X^a and weight tensor X^w, respectively. Thus, it is possible to exchange the computational order during network forward-propagation, as depicted in Equation (20), for better efficiency. With the exchange, the time-consuming convolutional computation is operated in integer format only (all elements $z^a \in Z^a$ and $z^w \in Z^w$ are b-bit integers). Therefore, benefits in aspects of latency, memory footprint and energy consumption can be achieved compared with corresponding floating-point counterpart

$$Q(X^a) \cdot Q(X^w) = (Z^a \cdot (2 \cdot Z^w - (l-1))) \cdot \left(\frac{\alpha^a \cdot \alpha^w}{(l-1)^2} \right). \tag{20}$$

In theory, for b-bit quantization network, the input activation and weight have a $\frac{32}{b}\times$ memory saving. For energy consumption, we list the estimated energy cost for per operation of different types on chip in Table 2. As we can see, the energy cost of floating-point ADD and $MULT$ is much larger than that of the fixed-point operation. Moreover, the DRAM access costs magnitude-order higher energy compared to the ALU operations. Therefore, it is clear that the quantized model is potential to save considerable energy compared to the full precision counterpart.

In terms of inference latency, the actual speedup of quantized models against full precision counterparts is decided by the computational ability of fixed-point arithmetic versus floating-point arithmetic on the platform. Table 3 shows the operations per cycle per SM on the Nvidia Turing architecture. We can learn from Table 3 that an 8-bit network is potential to achieve $2\times$ speedups versus the full precision counterpart on the platform. More impressively, 4-bit network and binary neural network (1-bit) are able to run faster than the full precision model by $4\times$ and $16\times$, respectively.

TABLE 2
Energy Consumption of Different Operations in 45nm CMOS Process

Operation	Energy (pJ)
32-bit Fixed-point ADD	0.1
32-bit floating-point ADD	0.9
32-bit Fixed-point MULT	3.1
32-bit floating-point MULT	3.7
32-bit 32KB SRAM	5
32-bit DRAM	640

4 Experiments

To evaluate the effectiveness of ABCNet v2, we conduct experiments on various scene text benchmarks, including multi-oriented scene text benchmarks ICDAR'15 [32], MSRA-TD500 [26], ReCTS [39], and two arbitrarily shaped benchmarks Total-Text [41] and SCUT-CTW1500 [42]. The ablation studies are conducted on Total-Text and SCUT-CTW1500 to verify each component of our proposed method.

4.1 Implementation Details

The backbone of the work here follows a common setting as most previous work, i.e., ResNet-50 [88] together with a Feature Pyramid Network (FPN) [89] unless specified otherwise. For the detection branch, we apply RoIAlign on 5 feature maps with 1/8, 1/16, 1/32, 1/64, and 1/128 resolution of the input image while for recognition branch, Bezier-Align is conducted on three feature maps with 1/4, 1/8, and 1/16 sizes, and the width and height of the sampling grid are set to 8 and 32, respectively. For English only dataset, the pretrained data is collected from publicly available English word-level-based datasets, including 150K synthesized data described in the next section, 7K ICDAR-MLT data [38], and the corresponding training data of each dataset. The pretrained model is then fine-tuned on the training set of the target datasets. Note that 15k COCO-Text [90] images in our previous manuscript [16] are not used in this improved version. For ReCTS dataset, we adopt LSVT [91], ArT [92], ReCTS [39], and the synthetic pretrained data to train the model.

In addition, we also adopt data augmentation strategies, e.g., random scale training, with the short size uniquely chosen from 640 to 896 (interval of 32) and the long size being less than 1600; and random crop, which we make sure that the crop image do not cut the text instance (for some special cases that hard to meet the condition, we do not apply random crop).

We train our model using 4 Tesla V100 GPUs with the image batch size of 8. The maximum iteration is 260K; and the initialized learning rate is 0.01, which reduces to 0.001 at the 160Kth iteration and 0.0001 at 220Kth iteration.

TABLE 3
Computational Ability (Ops per Cycle per SM) Comparison on Nvidia Turing Architecture

input precision	Output	Ops/Cycle/SM
FP16	FP16 or FP32	1024
INT8	INT32	2048
INT4	INT32	4096
INT1	INT32	16384

4.2 Benchmarks

*B*ezier Curve Synthetic Dataset 150k. For the end-to-end scene text spotting methods, a massive amount of free synthesized data are always necessary. However, the existing 800k SynText dataset [93] only provides a quadrilateral bounding box for a majority of straight text. To diversify and enrich the arbitrarily shaped scene text, we make some effort to synthesize a dataset of 150K images (94,723 images contain a majority of straight text, and 54,327 images contain mostly curved text) with the VGG synthetic method [93].

Specially, we filter out 40K text-free background images from COCO-Text [90] and then prepare the segmentation mask and scene depth of each background image for the following text rendering. To enlarge the shape diversity of synthetic texts, we modify the VGG synthetic method by synthesizing scene text with various art fonts and corpus and generate the polygonal annotation for all the text instances. The annotations are then used for producing the Bezier curve ground truth by the generating method described in Section 3.1.1. Examples of our synthesized data are shown in Fig. 8. For Chinese pretraining, we synthesized 100K images following the same method as above, with some examples shown in Fig. 9.

Total-Text dataset [41] is one of the most important arbitrarily shaped scene text benchmark proposed in 2017, which was collected from various scenes, including text-like scene complexity and low-contrast background. It contains 1,555 images, with 1,255 for training and 300 for testing. To resemble the real-world scenarios, most of the images of this dataset contain a large amount of regular text, while guarantee that each image has at least one curved text. The text instance is annotated with polygon based on word-level. Its extended version [41] improves its annotation of the training set by annotating each text instance with a fixed ten points following text recognition sequence. The dataset contains English text only. To evaluate the end-to-end results, we follow the same metric as previous methods, which use F-measure to measure the word-accuracy.

SCUT-CTW1500 dataset [42] is another important arbitrarily shaped scene text benchmark proposed in 2017. Compared to Total-Text, this dataset contains both English and Chinese text. In addition, the annotation is based on text-line level, and it also includes some document-like text, i.e., numerous small text may stack together. SCUT-CTW1500 contains 1k training images, and 500 testing images.

ICDAR 2015 dataset [32] provides images which are incidentally captured in the real world. Unlike previous ICDAR datasets, in which the text are clean, well-captured, and horizontally centered in the images. The dataset includes 1,000 training images and 500 testing images with complicate backgrounds. Some text may also appear in any orientation and any location, with small size or low resolution. The annotation is based on word-level, and it only includes English samples.

Fig. 7. *Qualitative results of ABCNet v2 on various datasets.* The detection results are shown with blue bounding boxes. Prediction confidence scores are also shown. Best viewed on screen.

Fig. 8. Examples of english Bezier curve synthesized data.

MSRA-TD500 dataset [26] contains 500 multi-oriented Chinese and English images, with 300 images for training and 200 images for testing. Most of the images are captured indoor. To overcome the insufficiency of the training data, we use the synthetic Chinese data mentioned above for model pretraining.

ICDAR'19-ReCTs dataset [39] contains 25k annotated signboard images, in which 20k images are for training set, and the rest are for testing set. Compared to English text, Chinese text normally has a significantly large number of the classes, with more than 6k commonly used characters with complicated layouts and various fonts. This dataset mainly contains text of the shop signs, and it also provides annotations for each character.

ICDAR'19-ArT dataset [92] is currently the largest dataset for arbitrarily shaped scene text. It is the combination and extension of the Total-text and SCUT-CTW1500. The new images also contains at least one arbitrarily-shaped text per image. There is a high diversity in terms of text orientations. The ArT dataset is split to a training set with 5,603 images and 4,563 for testing set. All the English and Chinese text instances are annotated with tight polygons.

ICDAR'19-LSVT dataset [91] provides an unprecedentedly large number of text from street view. It provides total 450k images with rich information of the real scene, among which 50k are annotated in full annotations (30k for training and the rest 20k for testing). Similar to ArT [92], this dataset also contains some curved text, which are annotated with polygon.

4.3 Ablation Study

To evaluate the effectiveness of the proposed components, we conduct ablation studies on two datasets, Total-Text and SCUT-CTW1500. We find that there are some training errors because of the different initialization. In addition, the text spotting task requires that all the characters should be correctly recognized. To avoid such issues, we train each method for three times and report the average results. The results are shown in Table 5, which demonstrate that all modules can lead to an obvious improvement against the baseline model on both datasets.

Fig. 9. Examples of Chinese Bezier curve synthesized data.

TABLE 4
Ablation Study for BezierAlign

Methods	Sampling method	F-measure (%)
baseline	+ Horizontal Sampling	38.4
baseline	+ Quadrilateral Sampling	44.7
baseline	+ BezierAlign	61.9

Horizontal sampling follows [3], and quadrilateral sampling follows [4].

We can see that using the attention-based recognition module, the results can be improved by 2.7 percent on Total-Text and 7.9 percent on SCUT-CTW1500, respectively.

We then evaluate all other modules using the attention-based recognition branch. Some conclusions are as follows:

- Using a biFPN architecture, the results can be improved by an additional 1.9 and 1.6 percent, while the inference speed is only reduced by 1 FPS. Thus, we achieve a better trade-off between speed and accuracy.
- Using coordinate convolution mentioned in Section 3.2, the results can be significantly improved by 2.8 and 2.9 percent on two datasets, respectively. Note that such improvement does not introduce noticeable computation overhead.
- We also test the AET strategy mentioned in Section 3.5, which results in 1.2 and 1.7 percent improvement.
- Lastly, we conduct experiments to show how the setting of Bezier curve order affects the results. Specifically, we regenerate all the ground truths for the same synthetic and the real images using 4th-order Bezier curves. We then train the ABCNet v2 by regressing the control points and use 4th-order BezierAlign. Other parts remain the same as the 3rd-order setting. The results shown in Table 5 demonstrate that increasing the order can be conducive to the text spotting results, especially on SCUT-CTW1500 which adopts text-line annotation. We further conduct experiments by using 5th-order Bezier curves on Total-text dataset following the same experimental setting; however, compared with baseline, we find that the performance drops from 66.2 to 65.5 percent in terms of the E2E Hmean. Based on the observation, we assume that the decline might be because an extremely higher order may result in drastic variation of the controlled points, which could exacerbate the difficulties of the regression. Some results using 4th-order Bezier curve are shown in Fig. 10. We can see that the detection bounding box can be more compact, and thus the textual feature can be more accurately cropped for subsequent recognition.

We further evaluate BezierAlign by comparing it with previous sampling methods, shown in Fig. 5. For fair and fast comparison, we use a small training and testing scale. The results shown in Table 4 demonstrate that the BezierAlign can dramatically improve the end-to-end results. Qualitative examples are shown in Fig. 11. Another ablation study is conducted to evaluate the time consumption of Bezier curve detection, and we observe that the Bezier curve detection only introduces negligible computation overhead compared with standard bounding box detection.

TABLE 5
Ablation Study on Both Total-Text and SCUT-CTW1500 Datasets

	Components					Total-Text			SCUT-CTW1500	
	Attn	biFPN	CoordConv	AET	4O	E2E Hmean	Impr.	FPS	E2E Hmean	Impr.
baseline (ctc) [16]						63.5	-	13	45.0	-
baseline+	✓					66.2	↑ **2.7%**	11	52.9	↑ **7.9%**
baseline+	✓	✓				68.1	↑ **4.6%**	10	54.5	↑ **9.5%**
baseline+	✓		✓			69.0	↑ **5.5%**	11	55.8	↑ **10.8%**
baseline+	✓			✓		67.4	↑ **3.9%**	11	54.6	↑ **9.6%**
baseline+	✓				✓	67.0	↑ **3.5%**	11	54.4	↑ **9.4%**
ABCNet v2	✓	✓	✓	✓		70.4	↑ **6.9%**	10	57.5	↑ **12.5%**

Attn: attention recognition model. 4O: using 4th order Bezier Curve.

4.4 Comparison With State-of-the-Art

We compare our method to previous methods on both detection and end-to-end text spotting tasks. An optimal setting including inference thresholds and testing scale is decided using grid search. For the detection task, we conduct experiments on four datasets including two arbitrarily-shaped datasets (Total-Text and SCUT-CTW1500), two multi-oriented datasets (MSRA-TD500, and ICDAR 2015) and one bilingual dataset ReCTS. The results in Table 6 demonstrate that our method can achieve state-of-the-art performance on all four datasets, outperforming previous state-of-the-art methods.

For end-to-end scene text spotting tasks, *ABCNet v2 achieves the best performance on SCUT-CTW1500 and ICDAR 2015 datasets, significantly outperforming previous methods*. The results are shown in Table 7. Although our method is worse than Mask TextSpotter [10] in terms of 1-NED on the ReCTS dataset, we argue that we do not use the provided character-level bounding box and ours shows clear advantages in terms of the inference speed. On the other hand, ABCNet v2 can still achieve better detection performance compared to Mask TextSpotter [10] according to Table 6.

Qualitative results of the test sets are shown in Fig. 7. From the figure, we can see that ABCNet v2 achieves a powerful recall ability for various text including horizontal, multi-oriented, and curved text, or long and dense text presentation styles.

4.5 Comprehensive Comparison With Mask TextSpotter v3

Comparison Using Few Data. We find out the proposed method can achieve a promising spotting result using only a small amount of training data. To validate the effectiveness, we use the official code of the Mask TextSpotter v3 [77], and conduct experiments following the same setting by training the model with only the official training data of TotalText. Specifically, the optimizer and the learning rate (0.002) of our method are set to the same as that of Mask TextSpotter v3. The batch sizes are set to 4, and both methods are trained with 230K iterations. To ensure the best setting of the both methods, Mask TextSpotter v3 is trained by minimum sizes of 800, 1000, 1200, and 1400, and maximum size of 2333. Testing is conducted with a minimum size of 1000 and maximum size of 4000. Our method is trained by a minimum size from 640 to 896 with the interval of 32, and the maximum size is set to 1600. To stabilize the training, AET strategy is not used for the few-shot training. Testing is conducted with a minimum size of 1000, and maximum size of 1824. We also conduct grid search to find the best threshold for the Mask TextSpotter v3. The results of different iterations are shown in Fig. 12. We can see that although

Fig. 10. *Comparison between cubic Bezier curve (left) and 4th-order Bezier curve (right).* There are some slight differences.

Fig. 11. *Qualitative recognition results of the quadrilateral sampling method and BezierAlign.* Left: original image. Top right: results by using quadrilateral sampling. Bottom right: results by using BezierAlign.

TABLE 6
Detection Results on Total-Text, SCUT-CTW1500, MSRA-TD500, and ICDAR 2015 Datasets

Methods	Total-Text			SCUT-CTW1500			MSRA-TD500			ICDAR 2015			ReCTS		
	R	P	H	R	P	H	R	P	H	R	P	H	R	P	H
Seglink [33]	30.3	23.8	26.7	48.4	38.3	42.8	70.0	86.0	77.0	76.8	73.1	75.0	-	-	-
DMPNet [34]	-	-	-	61.7	63.9	62.7	-	-	-	68.2	73.2	70.6	-	-	-
CTD+TLOC [42]	74.5	82.7	78.4	85.3	67.9	75.6	73.9	83.2	78.3	77.1	84.5	80.6	-	-	-
TextSnake [47]	74.5	8.7	78.4	77.8	82.7	80.1	81.7	84.2	82.9	84.9	80.4	82.6	-	-	-
EAST [35]	50.0	36.2	42.0	49.7	78.7	60.4	67.4	87.3	76.1	78.3	83.3	80.7	73.7	74.3	74.0
He *et al*. [4]	-	-	-	-	-	-	61.0	71.0	69.0	-	-	-	-	-	-
DeepReg [80]	-	-	-	-	-	-	70.0	77.0	74.0	-	-	-	-	-	-
Textboxes++ [94]	-	-	-	-	-	-	-	-	-	78.5	87.8	82.9	-	-	-
LSE [81]	-	-	-	77.8	82.7	80.1	81.7	84.2	82.9	-	-	-	-	-	-
ATTR [52]	76.2	80.9	78.5	80.2	80.1	80.1	82.1	85.2	83.6	83.3	90.4	86.8	-	-	-
MSR [51]	73.0	85.2	78.6	79.0	84.1	81.5	76.7	87.4	81.7	-	-	-	-	-	-
TextDragon [9]	75.7	85.6	80.3	82.8	84.5	83.6	-	-	-	-	-	-	-	-	-
TextField [76]	79.9	81.2	80.6	79.8	83.0	81.4	75.9	87.4	81.3	80.1	84.3	82.4	-	-	-
PSENet-1s [46]	78.0	84.0	80.9	79.7	84.8	82.2	-	-	-	85.5	88.7	87.1	83.9	87.3	85.6
Seglink++ [48]	80.9	82.1	81.5	79.8	82.8	81.3	-	-	-	80.3	83.7	82.0	-	-	-
LOMO [49]	79.3	87.6	83.3	76.5	85.7	80.8	-	-	-	83.5	**91.3**	87.2	-	-	-
CRAFT [44]	79.9	87.6	83.6	81.1	86.0	83.5	78.2	88.2	82.9	84.3	89.8	86.9	-	-	-
PAN [45]	81.0	89.3	85.0	81.2	86.4	83.7	**83.8**	84.4	84.1	81.9	84.0	82.9	-	-	-
Mask TTD [95]	74.5	79.1	76.7	79.0	79.7	79.4	81.1	85.7	83.3	**87.6**	86.6	87.1	-	-	-
CounterNet [50]	83.9	86.9	85.4	**84.1**	83.7	83.9	-	-	-	86.1	87.6	86.9	-	-	-
DB [43]	82.5	87.1	84.7	80.2	86.9	83.4	77.7	76.6	81.9	82.7	88.2	85.4	-	-	-
Mask TextSpotter [10]	82.4	88.3	85.2	-	-	-	-	-	-	87.3	86.6	87.0	**88.8**	89.3	89.0
DRRN [53]	**84.9**	86.5	85.7	83.0	**85.9**	84.5	82.3	88.1	85.1	84.7	88.5	86.6	-	-	-
ABCNet v2	84.1	**90.2**	**87.0**	83.8	85.6	**84.7**	81.3	**89.4**	**85.2**	86.0	90.4	**88.1**	87.5	**93.6**	**90.4**

TABLE 7
End-to-End Text Spotting Results on Total-Text, and SCUT-CTW1500

Methods	Total-Text		SCUT-CTW1500		ICDAR 2015 End-to-End			ReCTS	FPS
	None	Full	None	Full	S	W	G	1-NED	
TextBoxes++ [94]	36.3	48.9	-	-	73.3	65.9	51.9	-	1.4
Mask TextSpotter'18 [6]	52.9	71.8	-	-	79.3	73.0	62.4	-	2.6
TextNet [75]	54.0	-	-	-	-	-	-	-	2.7
Li *et al*. [11]	57.8	-	-	-	-	-	-	-	1.4
Deep Text Spotter [74]	-	-	-	-	54.0	51.0	47.0	-	-
Mask TextSpotter'19 [10]	65.3	77.4	-	-	83.0	77.7	73.5	**67.8**	2.0
Qin *et al*. [7]	67.8	-	-	-	-	-	-	-	4.8
CharNet [8]	-	-	-	-	80.1	74.5	62.2	-	1.2
FOTS [5]	-	-	21.1	39.7	83.6	79.1	65.3	50.8	-
RoIRotate* [9]	-	-	38.6	70.9	82.5	79.2	65.4	-	-
TextDragon [9]	48.8	74.8	39.7	72.4	82.5	78.3	65.2	-	2.6
ABCNet [16]	64.2	75.7	45.2	74.1	-	-	-	-	**17.9**
Boundary TextSpotter [12]	-	-	-	-	79.7	75.2	64.1	-	-
Craft [96]	**78.7**	-	-	-	83.1	82.1	74.9	-	5.4
Mask TextSpotter v3 [77]	71.2	78.4	-	-	83.3	78.1	74.2	-	2.5
Feng *et al*. [97]	55.8	79.2	42.2	74.9	**87.3**	**83.1**	69.5	-	7.2
ABCNet v2	70.4	78.1	57.5	77.2	82.7	78.5	73.0	62.7	10
ABCNet v2*	73.5	**80.7**	**58.4**	**79.0**	83.0	80.7	**75.0**	65.4	-

** represents the results are from [9]. "None" represents lexicon-free. "Strong Full" represents that we use all the words appeared in the test set. "S", "W", and "G" represent recognition with "Strong", "Weak", and "Generic" lexicon, respectively. FPS is for reference only, as it can be varied from different settings and devices. Here * represents multi-scale results.*

Mask TextSpotter v3 converges faster at the beginning, the final result of our method is better (56.41 versus 53.82 percent).

Comparison Using Large-Scale Data. We also use sufficient training data for a more thorough comparison with Mask TextSpotter v3. Formally, we carefully train Mask TextSpotter v3 using the Bezier Curve Synthetic Dataset (150k), MLT (7k), and TotalText, which are exact the same as our method. The training scales, batch sizes, the number of iterations, and others are all set to the same as mentioned in the Section 4.1. Grid search is also used to find the best threshold for the Mask TextSpotter v3. The results are shown in Table 8, from which we can see that Mask TextSpotter v3 outperforms ABCNet v1 by 0.9 in terms of F-measure, and ABCNet v2 can outperform Mask TextSpotter v3 (65.1 versus 70.4 percent). The inference time is measured under the same testing scale

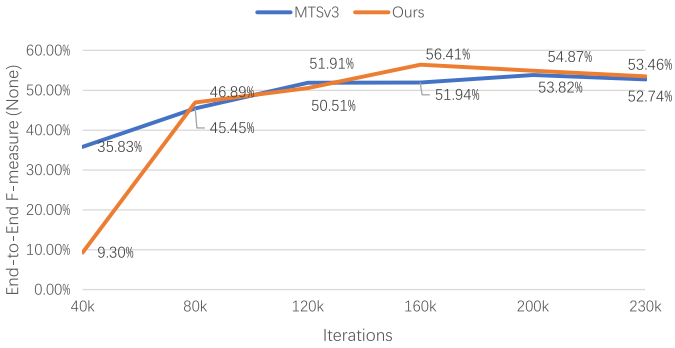


Fig. 12. Comparison with mask TextSpotter v3 using only the training set of TotalText. MTSv3: Mask TextSpotter v3 [77].

(Maximum size is set to 1824) and device (RTX A40), which further demonstrates the effectiveness of our method.

4.6 Limitations

We further conduct error analysis on the incorrectly predicted samples. We observe two types of common errors that may limit the scene text spotting performance of ABCNet v2.

The first one is shown in the example of Fig. 13. The text instance contains two characters. For each character, the reading order is from left to right. But for the whole instance, the reading order is from up to bottom. As the Bezier curve is interpolated in the longer side of the text instance, the Bezier-Align feature would be a rotated feature compared to its original feature, which can result in a completely different meaning. On the other hand, such cases only consist a minority of the whole training set, which is prone to be mistakenly recognized or predicted as an unseen category, as represented by "□" for the second character.

The second errors happen in different fonts, as shown in the middle of Fig. 13. The first two characters are written in unusual calligraphy fonts, making it difficult to recognize. In general, this challenge can only be alleviated with more training images.

We also find out that there is an extremely curved case in the test set of the CTW1500, where more than three crests exist in the same text instance, as shown in the third row of the Fig. 13. In such a case, the lower order such as cubic Bezier curve may be limited, as the character "i" is incorrectly recognized as the uppercase "I" because of the inaccurate shape representation. However, such cases are rarely seen, especially for those datasets using word-level bounding box.

4.7 Inference Speed

To further test the potential real-time performance of the proposed method, we exploit quantization techniques to significantly improve the inference speed. The baseline

TABLE 8
Comparison With Mask TextSpotter v3 Using Large-Scale Training Set

	MTSv3	ABCNet v1	ABCNet v2
F-measure (%)	65.1	64.2	**70.4**
FPS	2.5	**14.3**	8.7

MTSv3: Mask TextSpotter v3 [77].

Fig. 13. Error analysis of ABCNet v2.

model adopts the same setting as the baseline with an attention based recognition branch shown in Table 5. The backbone is replaced by ResNet-18, which significantly improves the speed with only marginal accuracy reduction.

The performance of the quantized network with various quantization bit configurations ($4/1$-bit) are reported in Table 9. Full precision performance is also listed for comparison. To train the quantized network, we first pretrained the low-bit model on the synthetic dataset. Then, we fine-tune the network on dedicated dataset $\mathit{TotalText}$ and $\mathit{CTW}1500$ for better performance.

Results of accuracy for both the pretrained model and fine-tuned model are reported. During the pre-training, 260K iterations are trained with batch size to be 8. Initial learning rate is set to be 0.01 and divided by 10 at 160K and 220K iteration. For the fine-tuning on $\mathit{TotalText}$ dataset, batch size remains 8 and the initial learning rate is set to be 0.001. Only 5K iterations are fine-tuned. The batch size and initial learning rate are the same when fine-tuning on $\mathit{CTW}1500$ dataset. However, the number of total iterations is set to be $120k$ and the learning rate is divided by 10 at iteration $80k$. Similar with previous quantization work, we quantize all the convolutional layer in the network, except input and output layers. If not specifically stated, networks are initialized with the full precision counterpart.

From Table 9, *we can learn that the 4-bit models by our quantization method are able to obtain comparable performance with the full precision counterparts.* For example, end-to-end hmean of 4-bit model pretrained on synthetic dataset is even better than that of the full precision model (57.6 versus 58.9 percent). After fine-tuning, end-to-end hmean of 4-bit model on $\mathit{TotalText}$ and $\mathit{CTW}1500$ is only 0.7 percent (67.2 versus 66.5 percent) and 0.8 percent (53.0 versus 52.2 percent) lower than that of the full precision model, respectively.

TABLE 9
Quantization Results of ABCNet v2 (ResNet-18 as the Backbone)

Data set	A/W	End-to-end Results			Detection only Results			FPS
		precision (%)	recall (%)	hmean (%)	precision (%)	recall (%)	hmean(%)	
Pretrain for Total-Text	32/32	66.1	51.0	57.6	83.3	76.7	79.9	16.4
Finetune on Total-Text	32/32	70.3	64.4	67.2	86.2	82.9	84.5	16.4
Finetune on CTW1500	32/32	58.8	48.3	53.0	88.2	81.4	84.7	37.9
Pretrain for Total-Text	4/4	67.2	52.4	58.9	83.4	77.7	80.5	25.9
Finetune on Total-Text	4/4	70.7	62.8	66.5	86.0	82.3	84.1	25.9
Finetune on CTW1500	4/4	58.0	47.3	52.2	87.2	81.0	84.0	47.8
Pretrain for Total-Text	1/1	62.0	34.4	44.3	82.0	63.6	71.6	29.5
Pretrain for Total-Text	1/1†	65.6	48.5	55.8	81.8	74.9	78.2	29.5
Finetune on Total-Text	1/1†	71.2	60.8	65.5	88.0	82.2	85.0	29.5
Finetune on CTW1500	1/1†	58.2	46.5	51.7	85.1	80.4	82.7	51.2

"A/W" indicates the bit width configuration of the activation and weight, respectively. † implies the case is trained with progressive training strategy. FPS is based on profiling data on single Nvidia 2080TI GPU.

The fact that almost no performance drops for 4-bit models (the same observation is also made on image classification and objection detection tasks [98]) indicates the considerable redundancy in the full precision scene text spotting model.

However, the performance has a sizable drop for the binary network, with the end-to-end hmean to be only 44.3 percent. To compensate, we propose to train the BNN (binary neural network) model with progressive training, in which the quantization bit width is progressively decreased (e.g., 4-bit → 2bit → 1-bit). With the new training strategy (the ones with † in the table), the performance of the binary network is significantly improved. For example, the end-to-end hmean trained on synthetic dataset by BNN model is only 1.8 percent (57.6 versus 55.8 percent) lower than the full precision counterpart.

Apart from the performance evaluation, we also compare the overall speed of the quantized model against full-precision models. In practice, only the quantized convolution layers are accelerated with other layers, such as LSTM keeping in full precision. It can be learned from Table 9 that, with limited performance drop, the binary network of ABCNet v2 is able to run in real-time for both TotalText and CTW1500 datasets.

5 Conclusion

We have proposed ABCNet v2—a real-time end-to-end method that uses Bezier curves for arbitrarily-shaped scene text spotting. By reformulating arbitrarily-shaped scene text using parameterized Bezier curves, ABCNet v2 can detect arbitrarily-shaped scene text with Bezier curves. Our method introduces negligible computation cost compared with standard bounding box detection. With such Bezier curve bounding boxes, we can naturally connect a light-weight recognition branch via a new BezierAlign layer, which is critical for accurate feature extraction, especially for curved text instances.

Comprehensive experiments on various datasets demonstrate the effectiveness of the proposed components, including using the attention recognition module, biFPN structure, coordinate convolution, and a new adaptive end-to-end training strategy. Finally, we propose to apply quantization techniques for deploying our model for real-time tasks, showing the great potential for a wide range of applications.

Acknowledgments

This work was done when Yuliang Liu, Chunhua Shen, Tong He, Peng Chen, and Hao Chen were with The University of Adelaide, Australia. The work of Lianwen Jin and Congyu Liu were supported by National Natural Science Foundation of China (NSFC) under Grant 61936003 and Natural Science Foundation of Guangdong Province (GD-NSF) under Grant 2017A030312006.

References

[1] F. L. Bookstein, "Principal warps: Thin-plate splines and the decomposition of deformations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, no. 6, pp. 567–585, Jun. 1989.
[2] M. Jaderberg *et al.*, "Spatial transformer networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2015, pp. 2017–2025.
[3] H. Li, P. Wang, and C. Shen, "Towards end-to-end text spotting with convolutional recurrent neural networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 5238–5246.
[4] T. He, Z. Tian, W. Huang, C. Shen, Y. Qiao, and C. Sun, "An end-to-end textspotter with explicit alignment and attention," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 5020–5029.
[5] X. Liu, D. Liang, S. Yan, D. Chen, Y. Qiao, and J. Yan, "FOTS: Fast oriented text spotting with a unified network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 5676–5685.
[6] 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 *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 67–83.
[7] S. Qin, A. Bissacco, M. Raptis, Y. Fujii, and Y. Xiao, "Towards unconstrained end-to-end text spotting," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 4703–4713.
[8] X. Linjie, T. Zhi, H. Weilin, and R. S. Matthew, "Convolutional character networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 9125–9135.
[9] F. Wei, H. Wenhao, Y. Fei, Z. Xu-Yao, and C.-L. Liu, "TextDragon: An end-to-end framework for arbitrary shaped text spotting," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 9075–9084.
[10] M. Liao, P. Lyu, M. He, C. Yao, W. Wu, and X. Bai, "Mask TextSpotter: An end-to-end trainable neural network for spotting text with arbitrary shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 2, pp. 532–548, Feb. 2021.
[11] H. Li, P. Wang, and C. Shen, "Towards end-to-end text spotting in natural scenes," *IEEE Trans. Pattern Anal. Mach. Intell.*, to be published, doi: 10.1109/TPAMI.2021.3095916.
[12] H. Wang *et al.*, "All you need is boundary: Toward arbitrary-shaped text spotting," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 12160–12167.

[13] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and efficient object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 10781–10790.

[14] R. Liu *et al.*, "An intriguing failing of convolutional neural networks and the CoordConv solution," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 9605–9616.

[15] X. Wang, R. Zhang, T. Kong, L. Li, and C. Shen, "SOLOv2: Dynamic and fast instance segmentation," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 17721–17732.

[16] Y. Liu, H. Chen, C. Shen, T. He, L. Jin, and L. Wang, "ABCNet: Real-time scene text spotting with adaptive Bezier-curve network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 9806–9815.

[17] S. M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, and R. Young, "ICDAR 2003 robust reading competitions," in *Proc. Int. Conf. Document Anal. Recognit.*, 2003, pp. 682–687.

[18] A. Shahab, F. Shafait, and A. Dengel, "ICDAR 2011 robust reading competition challenge 2: Reading text in scene images," in *Proc. Int. Conf. Document Anal. Recognit.*, 2011, pp. 1491–1496.

[19] D. Karatzas *et al.*, "ICDAR 2013 robust reading competition," in *Proc. IAPR Int. Conf. Document Anal. Recognit.*, 2013, pp. 1484–1493.

[20] B. Epshtein, E. Ofek, and Y. Wexler, "Detecting text in natural scenes with stroke width transform," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2010, pp. 2963–2970.

[21] W. Huang, Z. Lin, J. Yang, and J. Wang, "Text localization in natural images using stroke feature transform and text covariance descriptors," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 1241–1248.

[22] W. Huang, Y. Qiao, and X. Tang, "Robust scene text detection with convolution neural network induced MSER trees," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 497–511.

[23] G. Liang, P. Shivakumara, T. Lu, and C. L. Tan, "Multi-spectral fusion based approach for arbitrarily oriented scene text detection in video images," *IEEE Trans. Image Process.*, vol. 24, no. 11, pp. 4488–4501, Nov. 2015.

[24] M. Liao, B. Shi, X. Bai, X. Wang, and W. Liu, "TextBoxes: A fast text detector with a single deep neural network," in *Proc. AAAI Conf. Artif. Intell.*, 2017, pp. 4161–4167.

[25] Z. Tian, W. Huang, T. He, P. He, and Y. Qiao, "Detecting text in natural image with connectionist text proposal network," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 56–72.

[26] C. Yao, X. Bai, W. Liu, Y. Ma, and Z. Tu, "Detecting texts of arbitrary orientations in natural images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 1083–1090.

[27] R. Nagy, A. Dicker, and K. Meyer-Wegener, "NEOCR: A configurable dataset for natural image text recognition," in *Proc. Int. Workshop Camera-Based Document Anal. Recognit.*, 2011, pp. 150–163.

[28] X.-C. Yin, W.-Y. Pei, J. Zhang, and H.-W. Hao, "Multi-orientation scene text detection with adaptive clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1930–1937, Sep. 2015.

[29] P. Shivakumara, T. Q. Phan, and C. L. Tan, "A laplacian approach to multi-oriented text detection in video," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 2, pp. 412–419, Feb. 2011.

[30] X.-C. Yin, X. Yin, K. Huang, and H.-W. Hao, "Robust text detection in natural scene images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 5, pp. 970–983, May 2014.

[31] Z. Zhang, C. Zhang, W. Shen, C. Yao, W. Liu, and X. Bai, "Multi-oriented text detection with fully convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 4159–4167.

[32] D. Karatzas *et al.*, "ICDAR 2015 competition on robust reading," in *Proc. IAPR Int. Conf. Document Anal. Recognit.*, 2015, pp. 1156–1160.

[33] B. Shi, X. Bai, and S. Belongie, "Detecting oriented text in natural images by linking segments," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 3482–3490.

[34] Y. Liu and L. Jin, "Deep matching prior network: Toward tighter multi-oriented text detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 3454–3461.

[35] X. Zhou *et al.*, "EAST: An efficient and accurate scene text detector," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2642–2651.

[36] H. Hu, C. Zhang, Y. Luo, Y. Wang, J. Han, and E. Ding, "WordSup: Exploiting word annotations for character based text detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 4940–4949.

[37] B. Shi *et al.*, "ICDAR2017 competition on reading chinese text in the wild (RCTW-17)," in *Proc. IAPR Int. Conf. Document Anal. Recognit.*, 2017, pp. 1429–1434.

[38] N. Nayef *et al.*, "ICDAR2019 robust reading challenge on multi-lingual scene text detection and recognition–RRC-MLT-2019," in *Proc. IAPR Int. Conf. Document Anal. Recognit.*, 2019, pp. 1582–1587.

[39] R. Zhang *et al.*, "ICDAR 2019 robust reading challenge on reading chinese text on signboard," in *Proc. IAPR Int. Conf. Document Anal. Recognit.*, 2019, pp. 1577–1581.

[40] A. Risnumawan, P. Shivakumara, C.-S. Chan, and C. Tan, "A robust arbitrary text detection system for natural scene images," *Expert Syst. Appl.*, vol. 41, pp. 8027–8048, 2014.

[41] C.-K. Ch'ng, C. S. Chan, and C.-L. Liu, "Total-Text: Toward orientation robustness in scene text detection," *Int. J. Document Anal. Recognit.*, vol. 23, pp. 31–52, 2020.

[42] Y. Liu, L. Jin, S. Zhang, C. Luo, and S. Zhang, "Curved scene text detection via transverse and longitudinal sequence connection," *Pattern Recognit.*, vol. 90, pp. 337–345, 2019.

[43] M. Liao, Z. Wan, C. Yao, K. Chen, and X. Bai, "Real-time scene text detection with differentiable Binarization," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 11474–11481.

[44] Y. Baek, B. Lee, D. Han, S. Yun, and H. Lee, "Character region awareness for text detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 9357–9366.

[45] W. Wang *et al.*, "Efficient and accurate arbitrary-shaped text detection with pixel aggregation network," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 8439–8448.

[46] W. Wang *et al.*, "Shape robust text detection with progressive scale expansion network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 9328–9337.

[47] S. Long, J. Ruan, W. Zhang, X. He, W. Wu, and C. Yao, "TextSnake: A flexible representation for detecting text of arbitrary shapes," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 20–36.

[48] J. Tang, Z. Yang, Y. Wang, Q. Zheng, Y. Xu, and X. Bai, "SegLink++: Detecting dense and arbitrary-shaped scene text by instance-aware component grouping," *Pattern Recognit.*, vol. 96, 2019, Art. no. 106954.

[49] C. Zhang *et al.*, "Look more than once: An accurate detector for text of arbitrary shapes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 10544–10553.

[50] Y. Wang, H. Xie, Z.-J. Zha, M. Xing, Z. Fu, and Y. Zhang, "ContourNet: Taking a further step toward accurate arbitrary-shaped scene text detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11750–11759.

[51] C. Xue, S. Lu, and W. Zhang, "MSR: Multi-scale shape regression for scene text detection," in *Proc. Int. Joint Conf. Artif. Intell.*, 2019, pp. 989–995.

[52] X. Wang, Y. Jiang, Z. Luo, C.-L. Liu, H. Choi, and S. Kim, "Arbitrary shape scene text detection with adaptive text region representation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 6442–6451.

[53] S.-X. Zhang *et al.*, "Deep relational reasoning graph network for arbitrary shape text detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 9696–9705.

[54] L. Neumann and J. Matas, "Real-time scene text localization and recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 3538–3545.

[55] C. Yao, X. Bai, B. Shi, and W. Liu, "Strokelets: A learned multi-scale representation for scene text recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 4042–4049.

[56] B. Su and S. Lu, "Accurate scene text recognition based on recurrent neural network," in *Proc. Asian Conf. Comput. Vis.*, 2014, pp. 35–48.

[57] B. Su and S. Lu, "Accurate recognition of words in scenes without character segmentation using recurrent neural network," *Pattern Recognit.*, vol. 63, pp. 397–405, 2017.

[58] 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," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 11, pp. 2298–2304, Nov. 2017.

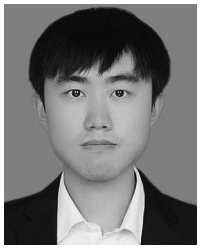
[59] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2006, pp. 369–376.

[60] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Proc. Int. Conf. Learn. Representations*, 2015. [Online]. Available: https://nyuscholars.nyu.edu/en/publications/neural-machine-translation-by-jointly-learning-to-align-and-trans-2

[61] F. L. Bookstein, "Thin-plate splines and the decompositions of deformations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, no. 6, pp. 567–585, Jun. 1989.

[62] B. Shi, X. Wang, P. Lyu, C. Yao, and X. Bai, "Robust scene text recognition with automatic rectification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 4168–4176.

[63] B. Shi, M. Yang, X. Wang, P. Lyu, C. Yao, and X. Bai, "ASTER: An attentional scene text recognizer with flexible rectification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 9, pp. 2035–2048, Sep. 2019.
[64] C. Luo, L. Jin, and Z. Sun, "MORAN: A multi-object rectified attention network for scene text recognition," *Pattern Recognit.*, vol. 90, pp. 109–118, 2019.
[65] W. Liu, C. Chen, and K.-Y. K. Wong, "Char-Net: A character-aware neural network for distorted scene text recognition," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 7154–7161.
[66] F. Zhan and S. Lu, "ESIR: End-to-end scene text recognition via iterative image rectification," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2054–2063.
[67] R. Litman, O. Anschel, S. Tsiper, R. Litman, S. Mazor, and R. Manmatha, "SCATTER: Selective context attentional scene text recognizer," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11959–11969.
[68] Z. Cheng, Y. Xu, F. Bai, Y. Niu, S. Pu, and S. Zhou, "AON: Towards arbitrarily-oriented text recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 5571–5579.
[69] H. Li, P. Wang, C. Shen, and G. Zhang, "Show, attend and read: A simple and strong baseline for irregular text recognition," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 8610–8617.
[70] X. Yue, Z. Kuang, C. Lin, H. Sun, and W. Zhang, "RobustScanner: Dynamically enhancing positional clues for robust text recognition," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp 135–151.
[71] M. Liao *et al.*, "Scene text recognition from two-dimensional perspective," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 8714–8721.
[72] Z. Wan, M. He, H. Chen, X. Bai, and C. Yao, "TextScanner: Reading characters in order for robust scene text recognition," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 12120–12127.
[73] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2015, pp. 91–99.
[74] M. Busta, L. Neumann, and J. Matas, "Deep TextSpotter: An end-to-end trainable scene text localization and recognition framework," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2223–2231.
[75] Y. Sun, C. Zhang, Z. Huang, J. Liu, J. Han, and E. Ding, "TextNet: Irregular text reading from images with an end-to-end trainable network," in *Proc. Asian Conf. Comput. Vis.*, 2018, pp. 83–99.
[76] Y. Xu, Y. Wang, W. Zhou, Y. Wang, Z. Yang, and X. Bai, "TextField: Learning a deep direction field for irregular scene text detection," *IEEE Trans. Image Process.*, vol. 22, no. 11, pp. 5566–5579, Nov. 2019.
[77] M. Liao, G. Pang, J. Huang, T. Hassner, and X. Bai, "Mask TextSpotter v3: Segmentation proposal network for robust scene text spotting," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 706–722.
[78] Z. Zhong, L. Sun, and Q. Huo, "An anchor-free region proposal network for faster R-CNN-based text detection approaches," *Int. J. Document Anal. Recognit.*, vol. 22, no. 3, pp. 315–327, 2019.
[79] Z. Tian, C. Shen, H. Chen, and T. He, "FCOS: Fully convolutional one-stage object detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 9626–9635.
[80] W. He, X.-Y. Zhang, F. Yin, and C.-L. Liu, "Deep direct regression for multi-oriented scene text detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 745–753.
[81] Z. Tian *et al.*, "Learning shape-aware embedding for scene text detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4229–4238.
[82] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2980–2988.
[83] J. Donahue *et al.*, "Long-term recurrent convolutional networks for visual recognition and description," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 677–691.
[84] T. Wang *et al.*, "Decoupled attention network for text recognition," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 12216–12224.
[85] R. J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neural Comput.*, vol. 1, no. 2, pp. 270–280, 1989.
[86] S. K. Esser, J. L. McKinstry, D. Bablani, R. Appuswamy, and D. S. Modha, "Learned step size quantization," in *Proc. Int. Conf. Learn. Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=rkgO66VKDS
[87] J. Choi, Z. Wang, S. Venkataramani, P. I.-J. Chuang, V. Srinivasan, and K. Gopalakrishnan, "PACT: Parameterized clipping activation for quantized neural networks," 2018, *arXiv:1805.06085*.
[88] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 630–645.
[89] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 936–944.
[90] A. Veit, T. Matera, L. Neumann, J. Matas, and S. Belongie, "COCO-text: Dataset and benchmark for text detection and recognition in natural images," 2016, *arXiv:1601.07140*.
[91] Y. Sun *et al.*, "ICDAR 2019 competition on large-scale street view text with partial labeling–RRC-LSVT," in *Proc. IAPR Int. Conf. Document Anal. Recognit.*, 2019, pp. 1557–1562.
[92] C.-K. Chng *et al.*, "ICDAR2019 robust reading challenge on arbitrary-shaped text (RRC-ArT)," in *Proc. IAPR Int. Conf. Document Anal. Recognit.*, 2019, pp. 1571–1576.
[93] A. Gupta, A. Vedaldi, and A. Zisserman, "Synthetic data for text localisation in natural images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2315–2324.
[94] M. Liao, B. Shi, and X. Bai, "TextBoxes++: A single-shot oriented scene text detector," *IEEE Trans. Image Process.*, vol. 27, no. 8, pp. 3676–3690, Aug. 2018.
[95] Y. Liu, L. Jin, and C. Fang, "Arbitrarily shaped scene text detection with a mask tightness text detector," *IEEE Trans. Image Process.*, vol. 29, no. 4, pp. 2918–2930, 2020.
[96] Y. Baek *et al.*, "Character region attention for text spotting," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 504–521.
[97] W. Feng, F. Yin, X.-Y. Zhang, W. He, and C.-L. Liu, "Residual dual scale scene text spotting by fusing bottom-up and top-down processing," *Int. J. Comput. Vis.*, vol. 129, pp. 619–637, 2021.
[98] R. Li, Y. Wang, F. Liang, H. Qin, J. Yan, and R. Fan, "Fully quantized network for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2805–2814.



Yuliang Liu received the BS degree from the South China University of Technology, Huizhou, China, and the PhD degree in information and communication engineering from the Deep Learning and Vision Computing Lab (DLVC-Lab), South China University of Technology, Huizhou, China, under the supervision of Prof. L. Jin. He was a postdoc supervised by Prof. Chunhua Shen with the University of Adelaide. His current research interests include computer vision, scene text detection, and recognition.

Chunhua Shen is a professor with the University of Adelaide, Australia.

Lianwen Jin (Member, IEEE) received the BS degree from the University of Science and Technology of China, Anhui, China, in 1991, and the PhD degree from the South China University of Technology, Guangzhou, China, in 1996. He is currently a professor with the College of Electronic and Information Engineering, South China University of Technology. His research interests include optical character recognition, computer vision, machine learning, and artificial intelligence. He has authored more than 200 scientific articles. He is a member of the IEEE Signal Processing Society and the IEEE Computer Society. He received the New Century Excellent Talent Program of MOE Award and the Guangdong Pearl River Distinguished Professor Award in 2006.

Tong He received the master's degree from Wuhan University, Wuhan, China, and the PhD degree from the University of Adelaide, Adelaide, Australia. He was a postdoc researcher with the University of Adelaide.

Peng Chen received the BS and PhD degrees from the University of Science and Technology of China, Hefei, China. He was a postdoc researcher with the University of Adelaide. His research interests include computer vision and machine learning algorithms, as well as model compression technologies.

Chongyu Liu received the BS degree in communication engineering from the Tianjin University, Tianjin, China, in 2018. He is currently working toward the PhD degree in information and communication engineering in the Deep Learning and Vision Computing Lab (DLVC-Lab), South China University of Technology, Huizhou, China, under the supervision of professor L. Jin. His current research interests include computer vision, scene text detection, and recognition.

Hao Chen received the BS and master's degrees from Zhejiang University, Hangzhou, China, and the PhD degree from the School of Computer Science, University of Adelaide, Adelaide, Australia.