ASTS: A Unified Framework for Arbitrary Shape Text Spotting

Juhua Liu, Zhe Chen, Bo Du, *Senior Member, IEEE*, and Dacheng Tao, *Fellow, IEEE*

***Abstract*— Arbitrary shape text spotting remains a challenging computer vision task. In this paper, we propose an end-to-end trainable unified framework for arbitrary shape text spotting to overcome the limitations inherent in the existing methods. Specifically, we propose to perceive and understand text based on different levels of semantics, *i.e.*, holistic-, pixel- and sequence-level semantics, and then unify the recognized semantics for robust text spotting. To implement the framework, we customize the detection and mask branches of Mask R-CNN to explore both holistic- and pixel-level semantics for text recognition. According to the recognition results, the text spotting task can then be formulated in the two-dimensional feature space. Then, by feeding the two-dimensional feature maps into an additional text recognition branch, our framework further delivers one-dimensional sequence-level semantics for text recognition based on an attention-based sequence-to-sequence network. Finally, the results from all the three levels of semantics are merged as the final result. Therefore, our framework is capable of simultaneously recognizing texts from both the one- and two-dimensional perspectives, achieving highly comprehensive text recognition. In addition, because some existing datasets lack character-level annotations, the extensive descriptions of texts from our framework further allow us to use only word-level annotations as weak supervision for training a robust text spotting model. Experiments on ICDAR 2013, ICDAR 2015, and Total-Text show that our framework achieves state-of-the-art performance for both detection and recognition.**

***Index Terms*— Scene text spotting, arbitrary shape, Mask R-CNN, weak supervision.**

Manuscript received September 6, 2019; revised February 13, 2020 and March 17, 2020; accepted March 21, 2020. Date of publication April 3, 2020; date of current version April 28, 2020. This work was supported in part by the National Natural Science Foundation of China under Grant 61601335, Grant 61822113, and Grant 41871243, in part by Australian Research Council Project under Grant FL-170100117, in part by the National Key R&D of China under Grant 2018YFA060550, in part by the Natural Science Foundation of Hubei Province under Grant 2018CFA050, and in part by the Science and Technology Major Project of Hubei Province (Next-Generation AI Technologies) under Grant 2019AEA170. The numerical calculations in this article have been done on the supercomputing system in the Supercomputing Center of Wuhan University. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Charith Abhayaratne. *(Corresponding author: Bo Du.)*

Juhua Liu is with the School of Printing and Packaging, Institute of Artificial Intelligence, Wuhan University, Wuhan 430072, China, and also with the UBTECH Sydney Artificial Intelligence Centre, Faculty of Engineering and Information Technologies, School of Computer Science, The University of Sydney, Darlington, NSW 2006, Australia (e-mail: liujuhua@whu.edu.cn).

Zhe Chen and Dacheng Tao are with the UBTECH Sydney Artificial Intelligence Centre, Faculty of Engineering and Information Technologies, School of Computer Science, The University of Sydney, Darlington, NSW 2006, Australia (e-mail: zhe.chen1@sydney.edu.au; dacheng.tao@sydney.edu.au).

Bo Du is with the National Engineering Research Center for Multimedia Software, Institute of Artificial Intelligence, Wuhan University, Wuhan 430072, China, and also with the School of Computer Science, Wuhan University, Wuhan 430072, China (e-mail: remoteking@whu.edu.cn).

Digital Object Identifier 10.1109/TIP.2020.2984082

I. Introduction

SCENE text spotting, which aims to simultaneously detect and recognize text with arbitrary shapes in natural scene images, is attracting increasing attention in the computer vision community [1]. Although there has been significant progress in both text detection and text recognition, this task remains challenging because the detection and recognition results can be greatly affected by complex backgrounds, diversified font sizes, and irregular shapes [2]–[7].

Existing scene text spotting methods can be classified into two main categories, *i.e.*, stepwise methods and end-to-end methods. Stepwise methods consider text spotting as two separate tasks, namely, text detection and text recognition [8]–[14]. These methods first detect texts from original images and then feed the detected text regions into a text recognition module. In text detection, a classifier is used to classify the text candidates extracted by well-designed detectors as text or non-text. In text recognition, character sequences are predicted using an image-based sequence model pre-trained with some word-annotated datasets. This procedure is simple and effective but ignores the correlation and complementarity between the two tasks, thereby producing sub-optimal results.

Multi-task learning, which not only enhances computational efficiency but also improves individual task performance [15]–[17], has recently been exploited to propose end-to-end methods that perform text detection and recognition simultaneously. These methods generally outperform stepwise methods [18]–[22]. However, most previous works are limited to text shapes and focus on horizontal and oriented texts. Therefore, they are not suitable for arbitrary shape texts, especially the curved texts frequently encountered in real-world scenarios. Example results of different methods are shown in Fig. 1. In contrast to these methods, Mask TextSpotter [20] localizes and recognizes text by segmenting word and character instances based on Mask R-CNN [17], thereby spotting texts with arbitrary shapes. However, when texts are relatively small or seriously blurred, the detection branch of the Mask R-CNN framework suffers character detection failures and consequently recognition failures. In practice, Mask TextSpotter requires character-level annotations for better training, while labeling characters in natural scene images is both time-consuming and laborious. Some existing datasets even lack annotations for character-level instances, which can further reduce the performance of Mask TextSpotter.

In this paper, we propose an end-to-end trainable unified framework for arbitrary shape text spotting (ASTS) to

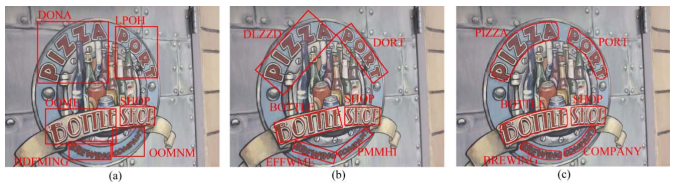


Fig. 1. Illustrations of the results of different text spotting methods: horizontal text spotting methods (a), oriented text spotting methods (b), and our proposed framework (c).

simultaneously detect and recognize texts of arbitrary shapes by analyzing three levels of semantics. Inspired by Mask R-CNN [17] and Mask TextSpotter [20], we can formulate text detection as an instance segmentation task. Unlike Mask TextSpotter, we customize the original detection and mask branches of Mask R-CNN to enable recognition at three levels, *i.e.*, holistic-, pixel- and sequence-level semantics. In particular, holistic- and pixel-level semantics can be used to detect texts and determine text shapes, respectively. In contrast to previous works, we concurrently implement text recognition from both the one- and two-dimensional perspectives. Specifically, in the detection and mask branches, we detect and segment not only word instances but also character instances. Therefore, we can implement text recognition via the result of instance segmentation in the two-dimensional feature space. Moreover, to better recognize text and tackle the problem of small or blurry texts, we add a text recognition branch with an attention-based optical character recognition (OCR) module based on Mask R-CNN to analyze sequential-level semantics. We employ the OCR module to predict the character sequence from the feature maps of word instances in an attentional sequence-to-sequence manner implemented in the one-dimensional feature space. Last, we fuse the two text recognition results according to the edit distance between the results and the given lexicon. Therefore, in our framework, texts can be comprehensively analyzed at all three levels of semantics, including the holistic level for the detection branch, pixel level for the mask branch, and sequential level for the recognition branch. The architecture of our framework is shown in Fig.2.

Meanwhile, to tackle the aforementioned issue of missing character-level annotations in some existing datasets, we also propose a weakly supervised learning scheme that trains a robust and accurate model by learning from word-level annotated datasets. Specifically, we first train a pre-trained model using our unified framework by learning on fully supervised datasets. Then, the pre-trained model is applied to the word-level annotated images to search for character samples and exploit word-level annotations to suppress false characters. Finally, the searched character samples can be used as character-level annotations and combined with word-level annotations to train a superior model.

In summary, the main contributions of this paper are as follows: 1) we propose an end-to-end trainable unified framework for arbitrary shape text spotting that concurrently detects and recognizes texts from natural scene images; 2) by performing recognition at three levels of semantics, our proposed framework is capable of recognizing texts from both the one- and two-dimensional perspectives simultaneously; 3) based on the proposed framework, we also introduce a weakly supervised learning scheme to obtain a more robust and accurate text spotting model using weakly supervised data; 4) our framework achieves state-of-the-art performance with respect to both text detection and recognition on several benchmark datasets, especially with irregular text.

The rest of this paper is organized as follows. In section II, we briefly review text detection, text recognition and text spotting. In section III, we introduce our proposed unified framework before analyzing each module. Section IV reports and discusses our experimental results. Finally, we conclude in section V.

II. Related Works

A. Text Detection

In stepwise text spotting methods, text detection directly impacts final recognition performance. Many text detection methods have been proposed, which can be classified into two main categories: character-based methods and word-based methods. The character-based methods regard text as composed of multiple characters [23]–[30]. They first extract characters with well-designed character detectors such as MSER [23], [24], SWT [25] and FASText [27] and then group these characters into words. However, character-based methods are time-consuming because their character detectors usually extract a huge number of candidates for character classification.

Benefiting from the development of common object detection, some word-based methods have been proposed to directly detect words in images [4]–[7]. Inspired by SSD [31], Liao *et al.* [4], [5] proposed TextBoxes and its extension TextBoxes++ by adding several text box layers. Shi *et al.* [7] employed the fully convolutional neural network (FCN) to detect text segments and their link relationships. Text segments are linked as the final detection result according to their relationship. In addition, some instance segmentation-based methods have been proposed to detect text in natural scene images [32]–[34]. Deng *et al.* [32] first segmented word instances by linking pixels within the same instance and extracting the bounding box of words from the segmentation result without location regression. Xie *et al.* [34] proposed a supervised pyramid context network (SPCNET) to locate text regions and suppress false positives based on Mask R-CNN. These instance segmentation-based methods usually detect text of arbitrary shape directly or through some minor modification.

B. Text Recognition

Text recognition methods can be classified into three main categories: character-based methods, word-based methods, and sequence-based methods. The character-based methods follow conventional OCR techniques, first segmenting and recognizing individual characters and then linking these characters into words [35], [36]. Word-based methods consider text

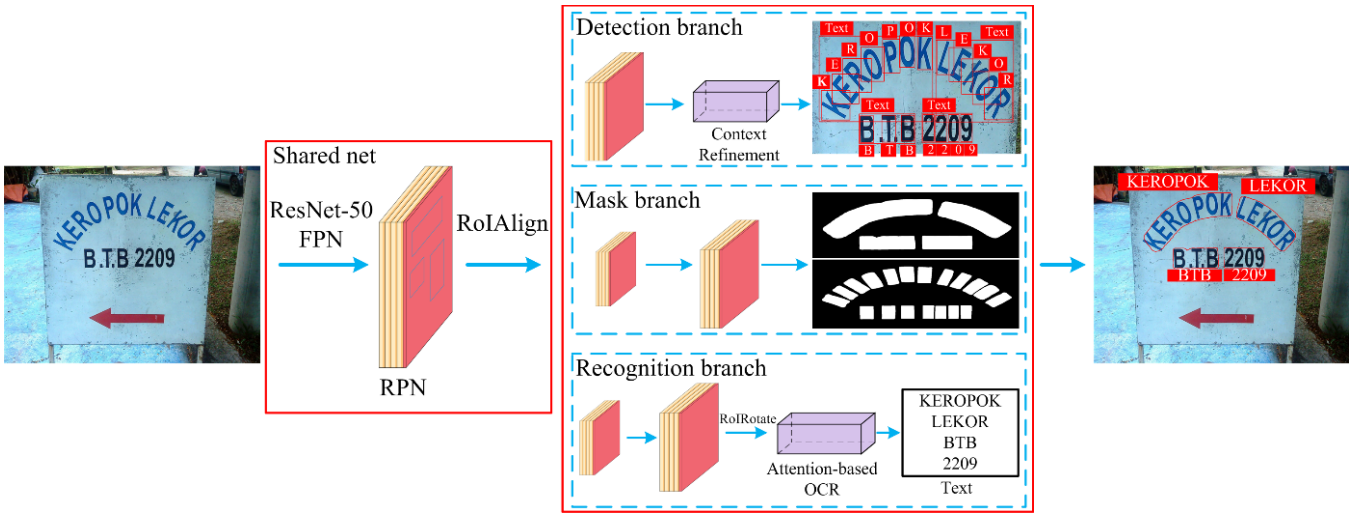


Fig. 2. The overall architecture of the proposed framework, which is in four parts: a shared network, text detection branch, text mask branch, and text recognition branch. The shared network extracts feature maps from the input image and shares them with the three subsequent branches. The text detection branch recognizes holistic-level semantics to detect character and word instances by refining region proposals. The text mask branch analyzes pixel-level semantics to conduct segmentation for the instances detected by the detection branch. The word masks are used to determine the text shapes, and the character masks are applied for preliminary recognition. The detection and mask branches implement text recognition from a two-dimensional perspective. The text recognition branch explores sequence-level semantics to recognize text from a one-dimensional perspective. Finally, the preliminary recognition results and the recognition branch outputs are merged as the final recognition results.

recognition as a word classification task, among which the most representative is the method of Jaderberg *et al.* [37]. Sequence-based methods treat text recognition as a sequence-to-sequence task. The input text image is first converted to a feature sequence via an encoder, and then connectionist temporal classification (CTC) or some other sequence-to-sequence decoder is used to predict the character sequence from the feature sequence. Shi *et al.* [14] used convolutional neural networks (CNNs) and recurrent neural networks (RNNs) to encode images to a feature sequence and employed CTC to decode the feature sequence to words. Lee *et al.* [38] employed several recursive convolutional layers to produce an input feature sequence for an attention-based RNN. According to the input features for recognition, character-based methods have been implemented from the two-dimensional perspective, while sequence-based methods have adopted the one-dimensional perspective.

C. Text Spotting

In most conventional text spotting methods, text spotting is considered in two separate stages: text detection and text recognition. These methods can be classified as stepwise methods; they usually first detect text with a pre-trained detection model and then use a separate recognition model for recognition. Inspired by R-CNN [39], Jaderberg *et al.* [10] detected text based on a region proposal mechanism and adopted a deep CNN for recognition. Based on a very large synthetic text image dataset, Gupta *et al.* [40] adopted a fully convolutional regression network (FCRN) and a word-based recognition method for text detection and recognition, respectively. Liao *et al.* [4], [5] used TextBoxes or its extension version TextBoxes++ for text detection and adopted a convolutional recurrent neural network (CRNN) [14] for recognition.

Recently, with the development of multi-task learning, some unified frameworks have been devised for text spotting. Busta *et al.* [18] first pre-trained detection and recognition CNNs separately and then trained both detection and recognition CNNs with a combined dataset. Therefore, this method cannot be trained in an end-to-end manner. Li *et al.* [19] proposed an end-to-end text spotter consisting, of a text proposal network (TPN) for text detection, an RNN encoder for embedding proposals of varying sizes to fixed-length vectors, and an attention-based RNN decoder for recognition. Liu *et al.* [22] proposed a fast oriented text spotting (FOTS) network that introduced RoIRotate to transform oriented feature maps to be horizontal for oriented text spotting. Most of these methods use relatively simple text/non-text information to train the text detection branch, which is probably inadequate for learning a discriminative representation. In addition, they are limited to the shape of the text and only applicable to horizontal and oriented text.

Our work is closely related to the recently proposed method of Lyu *et al.* [20], who devised Mask TextSpotter based on Mask R-CNN. Mask TextSpotter used the word instance segmentation result to determine text shape and employed the character instance for recognition. Mask TextSpotter directly implements recognition from the two-dimensional perspective. Our framework improves performance by adding a text recognition branch, which is specifically used to recognize small or seriously blurred text. Our text recognition branch includes an attention-based OCR module using a sequence-based recognition method to explore sequential-level semantics, and it is implemented from the one-dimensional perspective. Therefore, our framework implements text recognition from both the one- and two-dimensional perspectives, which complement each other and improve the final performance.

III. Methodology

In this section, we first introduce the pipeline of our proposed framework and then discuss details of the four main

modules: 1) the shared network for extracting image features and generating region proposals, 2) the detection branch for detecting word and character instances, 3) the mask branch for segmenting the results of the text detection branch, and 4) the recognition branch for recognizing text. We also explore the strategy of weakly supervised learning for optimizing our framework.

A. Overall Architecture

Fig. 2 depicts the overall architecture of our unified framework. First, an image is fed into the shared network, which uses a backbone network to extract feature maps from the input image and share them with the three subsequent branches via a region proposal network (RPN) and a RoIAlign [17] operation. Then, the text detection branch recognizes holistic-level semantics to detect and classify word and character instances by refining the region proposals using a context refinement network [41]. Meanwhile, the text mask branch analyzes pixel-level semantics to conduct segmentation for the instances detected by the detection branch. The word masks are used to determine the text shapes, and the character masks are applied for preliminary recognition. Since small or blurred characters are likely to suffer from detection failures, we feed two-dimensional feature maps of the word instances into the text recognition branch to explore sequence-level semantics for more precise recognition. Finally, the preliminary recognition results and the recognition branch outputs are merged according to the edit distance between the results and given lexicon.

B. Shared Network

The shared network consists of a backbone network and an RPN. Image features are extracted with the backbone and shared with subsequent branches. Feature pyramid networks (FPNs) [42] use a top-down architecture to build high-level feature maps at all scales from a single-scale input image and with marginal extra cost. Therefore, we use an FPN with ResNet-50 [43] as our backbone network.

The RPN is used to generate region proposals of various sizes and aspect ratios for the subsequent Faster R-CNN and mask branch in Mask R-CNN. In our framework, we adopt an RPN to generate the region proposals for these three subsequent branches. We assign anchors to different stages according to their sizes. In particular, in this paper, the areas of the anchors are set to $\{32^2, 64^2, 128^2, 256^2, 512^2\}$ pixels and assigned to five stages: $\{P2, P3, P4, P5, P6\}$, respectively. Different aspect ratios $\{0.5, 1, 2\}$ are used in each stage. Note that since RoIPooling [15] quantizes a floating number to the discrete granularity of the feature map, it could also lead to a misalignment problem. We instead apply RoIAlign [17] operation because it calculates image features based on coordinates in floating-point type with the bilinear interpolation method, thus delivering more accurate region alignments and more beneficial features to the subsequent branches.

C. Text Detection Branch

The text detection branch aims to refine the region proposals generated by the RPN by predicting their categories and adopting bounding box regression to adjust their coordinates. This processing can be formulated as the recognition of holistic-level semantics for the region proposals. The text detection branch usually needs to predict two categories: text or non-text. In contrast to most previous works [19], [22], our text detection branch aims to simultaneously detect both word and character instances. Thus, in addition to text/non-text information, we use extra character information to train the detection branch, which helps learn a more discriminative representation and improve detection performance. Moreover, according to the spatial relationship between word and character instances, the character detection results can be further used for text recognition. This is equivalent to implementing a character-based text recognition method while conducting text detection. Our text detection branch predicts K=64 categories including 26 capital letters, 26 lower case letters, 10 Arabic numerals, a word class, and a background class.

However, if a proposal partially overlaps with the true word, existing refinement methods may suffer refinement failure, since the proposal does not contain sufficient information for holistic object perception. Moreover, the OCR module is very sensitive to the text localization result. Therefore, we introduce a context refinement module [41] to augment the existing refinement in the text detection branch. From context refinement, the contextual information collected from regions surrounding the proposal is aggregated into a unified contextual representation to improve detection performance.

Following Fast R-CNN [15], the loss function of the text detection branch can be defined as

$$L_{det} = L_{cls}(p, u) + \lambda[u \geq 1]L_{loc}(t^u, v), \tag{1}$$

where $L_{cls}(p, u) = -logp_u$ is the log loss for true class u and p_u is the predicted confidence score for class u. L_{loc} is defined over a tuple of true bounding box regression targets for class u, v, and t^u is the predicted tuple. $\lambda[u \geq 1]$ means the output is 1 when $u \geq 1$ and 0 otherwise.

D. Text Mask Branch

The text mask branch aims to analyze pixel-level semantics and conduct semantic segmentation for word and character instances. The estimated word masks are used to provide more accurate word locations and shapes than the detection branch. Moreover, the character masks can be applied for recognition from the two-dimensional perspective. As shown in Fig. 3, we implement this branch with an FCN, the same as the mask branch in Mask R-CNN. Given an input region of interest (RoI) generated by the detection branch, RoIAlign extracts the features of the corresponding RoI from the shared feature maps and fixes their sizes to 14×14. Through four convolution layers and a deconvolution layer, the feature is up-scaled to 28×28. Then, a sigmoid function is applied to predict K=64 mask outputs including masks of the word instance, 62-character instances and the background.

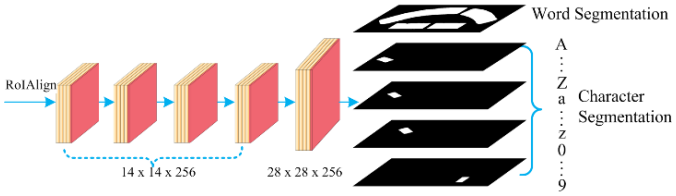


Fig. 3. Illustration of text mask branch.

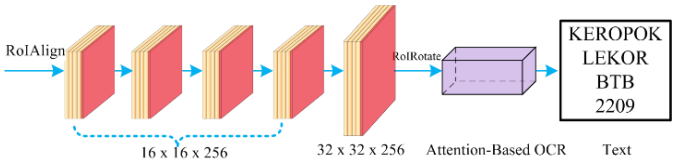


Fig. 4. Illustration of the text recognition branch.

Following Mask R-CNN [17], we also define the loss function of the text mask branch L_{mask} as an average binary cross-entropy loss. L_{mask} can be computed as follows:

$$L_{mask} = -\frac{1}{N}\sum_{n=1}^{N}[y_n \times \log(S(x_n)) + (1 - y_n) \times \log(1 - S(x_n))], \tag{2}$$

where N is the number of pixels, y_n is the pixel label ($y_n \in (0, 1)$), x_n is the predicted output, and $S(x)$ is the sigmoid function.

E. Text Recognition Branch

The text recognition branch aims to build an OCR module for exploring sequential-level semantics and predicting character sequences from the word instance feature maps. Text recognition can be treated as a sequence-to-sequence task. Moreover, the attention mechanism has also demonstrated the effectiveness of modeling dependencies without taking their distances in the input sequence into account. Therefore, we devise an attention-based OCR module for text recognition. Fig. 4 depicts the text recognition branch architecture, which consists of a small FCN and an attention-based OCR module. The resolution of each input feature map of this branch is 16×16. Like the previous mask branch, we also apply an FCN, which consists of four convolution layers and a deconvolution layer, to up-scale the size of feature maps to 32×32.

Since we need to convert the input two-dimensional feature map to a feature sequence before predicting the character sequence, we adopt RoIRotate [22] to transform the oriented feature maps to be horizontal before feeding them into the OCR module. For a given RoI, we obtain a rectangle with a rotation angle by calculating the minimum area rectangle of the mask generated by the mask branch. According to this rotation angle, we apply RoIRotate to transform the feature maps of RoI to be horizontal. Unlike the method in [22], in which feature maps are scaled to a fixed height and the aspect ratio is kept unchanged with padding, we keep the feature map fixed to 32×32. Although this inevitably distorts the image features, we argue that this has only minimal impact on the final performance since the same operation is used for both training and testing. Moreover, since the width of a feature map is relatively short, it is beneficial to reduce network oscillation in the training phase and make the network converge faster. We also conducted experiments with an unchanged aspect ratio, but the performance was worse.

Following the classic sequence-to-sequence model, we devise an attention-based OCR module consisting of an encoder and a decoder. The OCR module architecture is shown in Fig. 5, and the configuration is presented in Table I. The encoder converts feature maps into feature sequences,

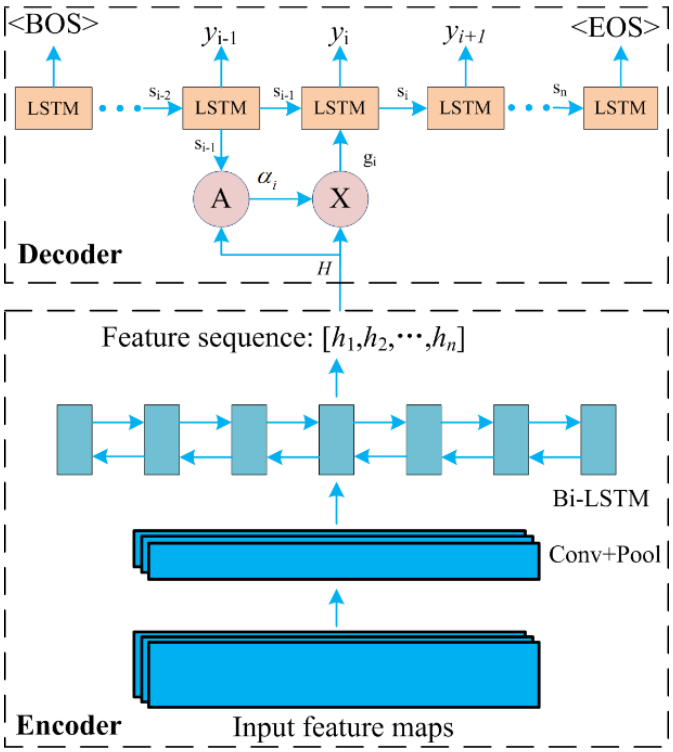


Fig. 5. The architecture of the attention-based OCR module.

TABLE I
THE CONFIGURATIONS OF THE ATTENTION-BASED OCR NETWORK

	Layer	Configurations
Encoder	Convolution	cn: 256, k: 3×3, s: 1×1, p: 1×1
	MaxPooling	k: 2×1, s: 2×1, p: 0×0
	Convolution	cn: 256, k: 3×3, s: 1×1, p: 1×1
	MaxPooling	k: 2×1, s: 2×1, p: 0×0
	Convolution	cn: 256, k: 3×3, s: 1×1, p: 1×1, bn
	Convolution	cn: 256, k: 3×3, s: 1×1, p: 1×1
	MaxPooling	k: 2×1, s: 2×1, p: 0×0
	Convolution	cn: 512, k: 3×3, s: 1×1, p: 1×1, bn
	Convolution	cn: 512, k: 3×3, s: 1×1, p: 1×1, bn
	MaxPooling	k: 2×1, s: 2×1, p: 0×0
	Convolution	cn: 512, k: 3×3, s: 1×1, p: 0×0, bn
	Bidirectional- LSTM	hidden units: 256
	Bidirectional- LSTM	hidden units: 256
Decoder	LSTM with attention	hidden units: 256 attention units: 256
	LSTM with attention	hidden units: 256 attention units: 256

"cn" represents the output channel, 'k', 's', and 'p' represent the kernel, stride and padding size, respectively, and "bn" represents batch normalization.

and the decoder aims to predict the character sequence from the feature sequence. Our encoder model is similar to the feature sequence extraction network in CRNN [14], except the input is the feature map in our encoder but the input image in CRNN. The encoder first converts the feature map to a feature sequence through 7 convolution and 4 max-pooling layers. Then, the feature sequence is fed into a multi-layer bidirectional LSTM to capture long-range dependencies of the feature sequence. Finally, the encoder feeds the output feature sequence into the decoder as the context. We denote the output feature sequence of the encoder as $H = [h_1, h_2, \cdots, h_n], n = 31$.

Based on the attention sequence-to-sequence model [44], we adopt an attention-based RNN to build our decoder, which aims to predict the result of character sequence $Y = [y_0, y_1, y_2, \cdots, y_T, y_{T+1}]$, where T is the length of the character sequence and y_0 and y_{T+1} represent the BOS token and EOS token, respectively. At the i-th step, the decoder predicts y_i by focusing on the relevant elements of H:

$$e_{i,j} = w^T tanh(Ws_{i-1} + Vh_j + b), \tag{3}$$

$$\alpha_{i,j} = exp(e_{i,j}) / \sum_{j=1}^{n} exp(e_{i,j}), \tag{4}$$

$$g_i = \sum_{j=1}^{n} \alpha_{i,j} h_j, \tag{5}$$

$$y_i \sim Generate(s_{i-1}, g_i), \tag{6}$$

where α_i is a vector of attention weights that can be computed by the encoder output H and the last state of RNN s_{i-1}. α_i indicates the weight of each encoder output when predicting the current character y_i. After predicting y_i, we also need to calculate the current state of RNN s_i, which is used to predict the next character y_{i+1}:

$$s_i = LSTM(s_{i-1}, g_i, y_i), \tag{7}$$

where $LSTM$ represents the recurrent activation function of unidirectional long short-term memory units (LSTM). Since y_i is incorporated into the calculation of the current state of RNN s_i, and s_i is used to predict the next character y_{i+1}, our OCR module implicitly contains a language model.

Let $X^t = \{x_0^t, x_1^t, x_2^t, \cdots, x_{T+1}^t\}$ be the ground truth tokens for word instance t and $Y^t = \{y_0^t, y_1^t, y_2^t, \cdots, y_{T+1}^t\}$ be the corresponding output sequence of the decoder. The recognition branch loss can be computed by

$$L_{recog} = -\frac{1}{N} \sum_{n=1}^{N} \sum_{i=1}^{T+1} \log y_i(x_i), \tag{8}$$

where N is the number of texts to be trained and $y_i(x_i)$ represents the predicted probability of the output being x_i at the i-th step.

Combined with the detection loss L_{det} in Eq. (1), the mask loss L_{mask} in Eq. (2), and the recognition loss L_{recog} in Eq. (8), the full multi-task loss function is

$$L = L_{rpn} + L_{det} + L_{mask} + L_{recog}, \tag{9}$$

where L_{rpn} is the loss of RPN, which is the same as in [16].

Our unified framework can detect and recognize texts by analyzing three levels of semantics, including holistic-, pixel-, and sequence-level semantics, and can also produce two recognition results. The first is the text spotting from the text detection and mask branches implemented from the two-dimensional perspective, and the other is the result of the text recognition branch from the one-dimensional perspective. The former result is more robust to inaccurate localization in the text detection branch, while the latter is relatively stable to font size and text quality. Therefore, aiming to improve recognition, we choose the word with the shortest edit distance from the given lexicon as the final recognition result.

F. Weakly Supervised Learning

As mentioned previously, some existing datasets only provide word-level annotations, and labeling characters is a time-consuming and laborious task. By recognizing texts comprehensively, the proposed framework allows us to achieve promising performance based on weak supervision. Inspired by the idea of weakly supervised learning [45], we introduce a weakly supervised learning scheme, aiming to train a more robust and accurate text spotting model by learning from the weakly supervised data with a pre-trained model. Specifically, we have a pre-trained model M and a weakly supervised dataset A that only has word-level annotations. The pre-trained model is trained using our proposed framework by learning from fully supervised datasets with both word- and character-level annotations. Each image in dataset A has weak annotations at the word level as denoted by a set of polygons $G = \{g_1, g_2, \cdots, g_i, \cdots\}$. The target of the weakly supervised learning is to search for character samples in dataset A.

An overview of the pipeline of our proposed weakly supervised learning is illustrated in Fig. 6. We first apply the pre-trained model M on the word-level annotated dataset A. For each image in dataset A, we can obtain a set of candidate samples R:

$$R = \{(c_0, s_0, b_0, m_0, t_0), (c_1, s_1, b_1, m_1, t_1), \cdots, (c_i, s_i, b_i, m_i.t_i), \cdots, \}, \tag{10}$$

where c_i, s_i, b_i, m_i, and t_i represent the predicted category, confidence score, bounding box, mask and recognition result of the i-th candidate character sample r_i, respectively.

Then, we can obtain the positive character samples with a confidence score threshold and the word-level annotation G:

$$P = \{(c_i, s_i) \mid c_i \in C \text{ and } s_i > S \text{ and } \frac{m_i \cap g_i}{m_i} > T\}, \tag{11}$$

where C denotes all character categories to be detected, S represents the confidence score threshold used to identify positive character samples, $m_i \cap g_j$ denotes the intersection overlap of candidate character sample r_i with its word-level ground truth g_j, and T is the threshold to select positive character samples. The confidence score threshold S can be set relatively lower due to the constraint provided by the word-level annotations, which is also beneficial for maintaining the diversity of the character samples. In this paper, S and T are set to 0.1 and 0.8, respectively.

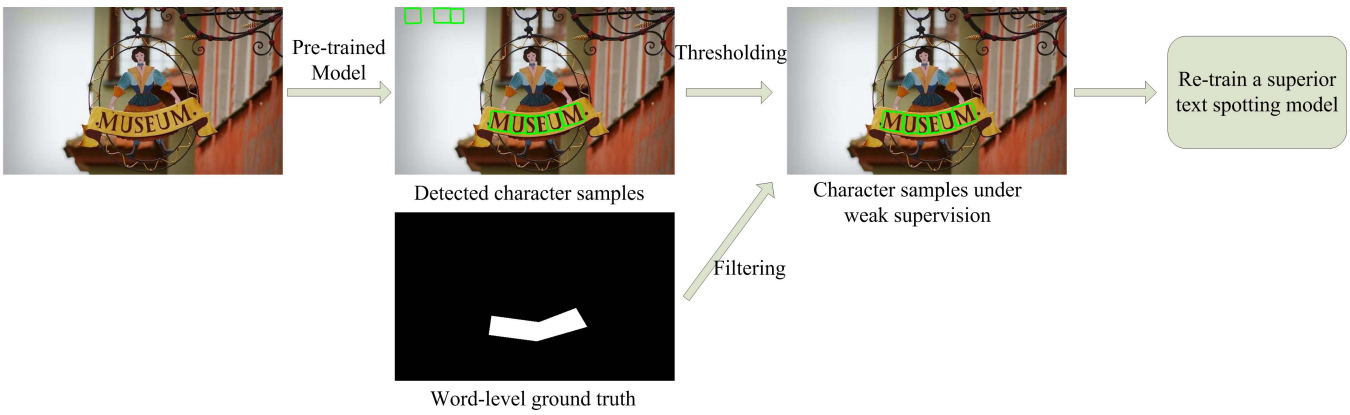


Fig. 6. Overview of our proposed weakly supervised learning pipeline. The results of applying the pre-trained model also contain word instances, but we only keep the character instances for visualization.

Finally, the identified positive character samples P can be used as character annotations and be combined with the word-level annotations G to re-train a more robust and accurate text spotting model.

IV. Experiments

In this section, we evaluate our framework by comparing it with state-of-the-art methods on several challenging public benchmark datasets. In sections A and B, we introduce the benchmark datasets and evaluation protocols used to conduct the experiments. The implementation details are described in section C. The results and comparisons on different datasets are listed in sections D, E and F, respectively. Further analysis is discussed in section G.

A. Datasets

In this paper, our goal is to detect and recognize text with arbitrary shape. Therefore, we conduct experiments on three public benchmark datasets: ICDAR 2013 [46], ICDAR 2015 [47] and Total-Text [48], which are horizontal, oriented and curved text datasets, respectively. Most texts in the three datasets are various fonts, sizes and colors. Moreover, some texts suffer from degradation, such as uneven light, blurring and overexposure.

ICDAR 2013 focuses on horizontal text detection and recognition and was proposed in Challenge 2 of the ICDAR 2013 Robust Reading Competition. It consists of a training set of 229 images and a test set of 233 images. In addition to word-level annotations, ICDAR 2013 also provides character-level annotations. All annotations are labeled with bounding boxes and corresponding transcriptions. In addition, for the test phase of text recognition, ICDAR 2013 provides three specific lexicons for each test image: strongly contextualized, weakly contextualized and generic. These lexicons can be used to correct the recognition results.

ICDAR 2015 was proposed in Challenge 4 of the ICDAR 2015 Robust Text Reading Competition. It focuses on incidental scene text detection and recognition. This dataset provides 1000 training images and 500 test images collected using Google smart glasses. Compared with ICDAR 2013, the texts in ICDAR 2015 are poor quality with more scales, blurring, orientation and other degradation. This dataset only contains word-level annotations, which are labeled with quadrangles and their corresponding transcriptions. Similar with ICDAR 2013, this dataset also provides strongly contextualized, weakly contextualized and general lexicons for each test image for correcting the recognition results.

Total-Text is a comprehensive dataset for scene text detection and recognition proposed by Ch'ng and Chan [48]. In addition to horizontal and oriented text, the dataset contains many curved texts. Total-Text contains 1255 training images and 300 test images. This dataset only provides word-level annotations with polygons and their corresponding transcriptions. Different from ICDAR 2013 and ICDAR 2015, this dataset does not provide any vocabulary for correcting recognition results in the test phase.

In addition to the above three datasets, we used **SynthText** [40] and **SCUT_FORU** [49] in the training phase. SynthText is a synthetically generated dataset proposed by Gupta *et al.* [40] and is usually used for pre-training text detection and recognition models. This dataset consists of 800,000 images with 8 million synthetic word instances, which are annotated at the word level and character level by the rotated rectangles and their corresponding transcriptions. SCUT_FORU is collected from the Flickr website and contains 1162 images. This dataset provides both character- and word-level bounding boxes, and we can obtain more precise word-level annotations by calculating the minimum area rectangle of all characters in the word.

B. Evaluation Protocol

Classic evaluation protocols adopt $recall$ (**R**), $precision$ (**P**), and $f-measure$ (**F**) as the evaluation metrics for text detection, word spotting and end-to-end recognition. They are given as follows:

$$R = \frac{TP}{TP + FN} \times 100\%, \tag{12}$$

$$P = \frac{TP}{TP + FP} \times 100\%, \tag{13}$$

$$F = 2 \times \frac{P \times R}{P + R} \times 100\%, \tag{14}$$

where TP, FP, and FN represent the numbers of true positives, false positives, and false negatives, respectively. For text detection, a detected region is considered a true positive if the intersection over union (IoU) between the detected polygon and the ground truth is larger than a given threshold (generally set to 0.5). The true positives in word spotting and end-to-end recognition require not only IoU restriction but also correct recognition results. Since there is a trade-off between $precision$ and $recall$, we adopt $f-measure$ as the evaluation criterion in this paper.

It should be noted that although we distinguish between uppercase and lowercase letters when recognizing text, we ignore the case according to the evaluation protocol.

C. Implementation Details

We implemented our framework based on the Maskrcnn-benchmark [50], and all experiments were conducted on a high-performance computing server with NVidia Tesla V100 (16G) GPUs. The model was trained with 4 GPUs and evaluated with 1 GPU.

Training: Different from some previous works [4], [5] [10], [40] that treat text detection and recognition as two separate tasks, we trained our proposed framework in an end-to-end manner. The training process was divided into two stages: pre-training on Synthtext and fine-tuning on a mixed real dataset. The mixed real dataset consisted of 229 training images from ICDAR 2013, 1000 training images from ICDAR 2015, 1255 training images from Total-Text, and 1162 images from SCUT_FORU. This allowed us to train a general text spotter model for all horizontal, oriented, and curved texts. It should be noted that some hardly recognized text regions in ICDAR 2015 and Total-Text are labeled as "DO NOT CARE", and we ignore them in the training phase.

Since there are only word-level annotations and no character-level annotations in ICDAR 2015 and Total-Text, we explored two fine-tuning schemes. The first scheme was to directly fine-tune on the mixed dataset with existing annotations, while the other fine-tuned on the combined annotations derived from the weakly supervised learning. For the weakly supervised learning, we applied the pre-trained model on ICDAR 2005 and Total-Text to search for character samples under weak supervision of their corresponding word-level annotations. The identified character samples were combined with their original word-level annotations to re-train a more robust and accurate model. We denoted these two models as "ASTS_Baseline" and "ASTS_Weakly", respectively.

Stochastic gradient descent (SGD) was adopted for model optimization. The weight decay was set to 0.0001, the momentum to 0.9, and the batch size to 8. During pre-training, we trained the model on SynthText for 15 epochs. The learning rate was set to 0.001 for the first 10 epochs and divided by 10 in the last 5 epochs. During fine-tuning, the training iterations were set to 100K; the learning rate was set to 0.0005 for the first 50K iterations and divided by 10 for the remainder.

Data Augmentation: To enhance network robustness, the following data augmentation strategies were applied. 1) Training with multi-scales: The short side of training images was scaled to one of three scales (600, 800 and 1000) randomly, and the maximum size of the longer side was set to 1280. Note that the aspect ratio of the image was maintained while the image was scaled. 2) Random rotation: The images were randomly rotated in a degree range of -15 to 15. 3) Random color adjustment: The hue, brightness, contrast, gamma and color mode of the images were adjusted randomly. 4) Random added noise: The images were degraded by adding salt-and-pepper noise randomly.

Inference: During inference, the shorter side of the test image was scaled to 1000 while keeping the aspect ratio unchanged. Then, the scaled image was fed into the shared network. The backbone network extracted image feature maps, and RPN generated region proposals for subsequent branches. The top 1,000 region proposals were input into the detection branch. After the classification and bounding box regression, NMS was adopted to suppress the redundant bounding boxes. The remaining regions were segmented in the mask branch. After the feature maps were rotated using RoIRotate, the rotated feature maps were recognized via our attention-based OCR module in the text recognition branch. Finally, the recognition results from all three branches were merged as the final result.

D. Results for ICDAR 2013

As mentioned above, ICDAR 2013 focuses on horizontal text, so we use it to evaluate the effectiveness of our method for detecting and recognizing horizontal text. Table II lists the results of our framework compared to some previous methods. Note that FPS is only for reference, since different GPUs are adopted in different methods.

As shown in Table II, compared with previous methods, both ASTS_Baseline and ASTS_Weakly achieve competitive performance for ICDAR 2013 detection and recognition. Relative to ASTS_Baseline, the performance of ASTS_Weakly is clearly improved, and it achieves state-of-the-art performance. The closest methods to our results for ASTS_Weakly are Mask TextSpotter and FOTS, where "FOTS" was implemented at the single scale while "FOTS_MS" represents the multi-scale inference result. For detection, ASTS_Weakly with single-scale inference outperforms the single-scale implementation by a relatively large margin (F-measure: 93.3 *vs*. 91.7) and outperforms FOTS_MS. For recognition, ASTS_Weakly also slightly exceeds or is equal to the previous works, except end-to-end recognition with a general lexicon. Our method is only slightly better than the other methods because the text in ICDAR 2013 is horizontal and most of the images are good quality, so previous works also perform well.

Fig. 7 shows some example results for ICDAR 2013. Some special types of degradation, such as uneven light (a), rare fonts (b) (e), small font-size text (c) and same-color background (d), do not affect the performance of our method. However, there are still some extreme cases (Fig. 8 (f), (g) and (h)) for which the method fails to detect or recognize text.

TABLE II
DETECTION AND RECOGNITION RESULTS FOR ICDAR 2013

Method	Year	Detection	FPS	Method	Year	Word Spotting			End-to-End			FPS
						S	W	G	S	W	G	
HUST_MCLAB [7]	2017	85.3	20.6	FCRNall+multi-filt [40]	2016	—	—	—	84.7	—	—	—
TextBoxes++_MS [5]	2018	89.0	—	TextBoxes [4]	2017	93.9	92.0	85.9	91.6	89.7	83.9	—
PixelLink [32]	2018	88.1	—	Li et al. [19]	2017	94.2	92.4	88.2	91.1	89.8	84.6	1.1
SPCNet [34]	2018	92.4	—	Deep TextSpotter [18]	2017	92.0	89.0	81.0	89.0	86.0	77.0	10
Mask TextSpotter [20]	2018	91.7	4.6	Mask TextSpotter [20]	2018	92.5	92.0	88.2	92.2	91.1	**86.5**	4.8
TextNet [21]	2018	91.4	2.7	TextNet [21]	2018	94.6	93.5	87.0	89.8	88.8	83.0	2.7
FOTS [22]	2018	88.3	**22**	FOTS [22]	2018	92.7	90.7	83.5	88.8	87.1	80.8	**22**
FOTS_MS [22]	2018	92.8	—	FOTS_MS [22]	2018	**95.9**	94	87.8	92.0	90.1	84.8	—
ASTS_Baseline	—	91.7	4.5	ASTS_Baseline	—	94.4	93.1	86.5	90.5	89.2	83.6	4.5
ASTS_Weakly	—	**93.5**	4.5	ASTS_Weakly	—	**95.9**	**94.7**	**88.5**	**92.8**	**91.5**	85.9	4.5

All methods of text detection are evaluated under the DetEval evaluation protocol. 'S', 'W', and 'G' represent recognition with strong, weak, and generic lexicons, respectively.

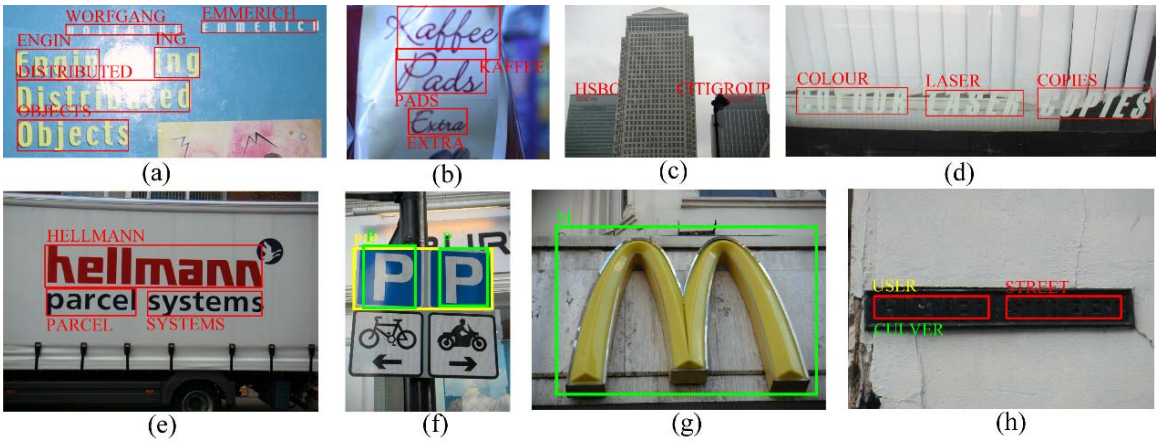


Fig. 7. Example results for ICDAR 2013. The texts and boxes in red represent texts and regions correctly spotted by ASTS. The texts and boxes in yellow represent texts and regions incorrectly spotted by ASTS. The texts and boxes in green represent the texts and regions of the ground truth. Images from (a) to (e) are some success example results, while images from (f) to (h) are failure example results.

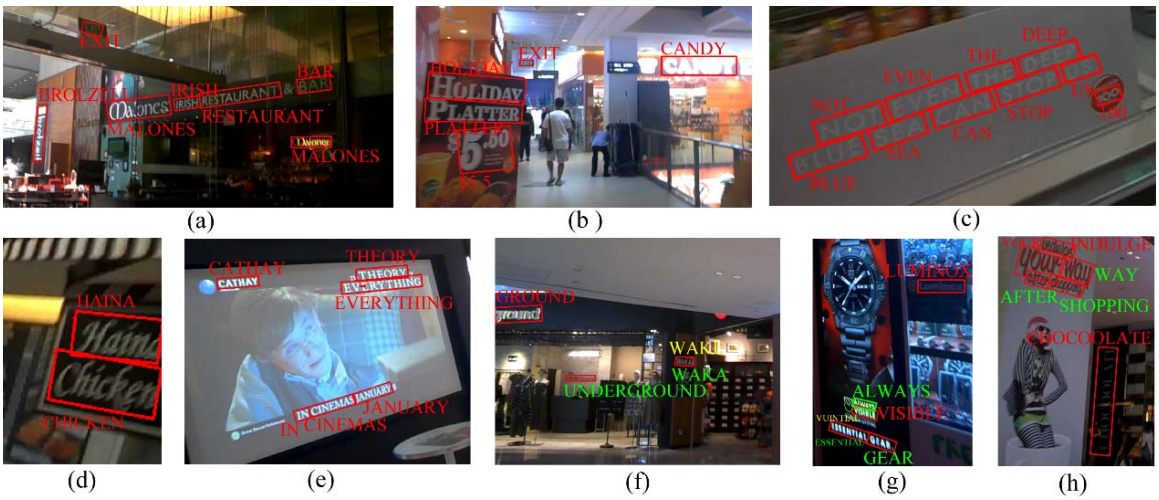


Fig. 8. Example results for ICDAR 2015. The texts and boxes in red represent texts and regions correctly spotted by ASTS. The texts and boxes in yellow represent texts and regions incorrectly spotted by ASTS. The texts and boxes in green represent the texts and regions of the ground truth. Images from (a) to (e) are some success example results, while images from (f) to (h) are failure example results.

E. Results for ICDAR 2015

We next evaluated the effectiveness of our method for detecting and recognizing oriented text on ICDAR 2015. Our results and comparison with previous methods are shown in Table III. Similar to ICDAR 2013, ASTS_Weakly achieves similar improvement based on ASTS_Baseline. Again, Mask TextSpotter and FOTS are the closest methods. "Mask TextSpotter(1000)" means Mask TextSpotter conducting inference with a single scale of 1000, while "Mask TextSpotter

TABLE III
DETECTION AND RECOGNITION RESULTS FOR ICDAR 2015

Method	Year	Detection	FPS	Method	Year	Word Spotting			End-to-End			FPS
						S	W	G	S	W	G	
HUST_MCLAB [7]	2017	75.0	—	Stradvision [47]	2015	45.9	—	—	43.7	—	—	—
TextBoxes++_MS [5]	2018	82.9	2.3	TextSpotter [13]	2016	37.0	21.0	16.0	35.0	20.0	16.0	1
PixelLink [32]	2018	83.7	3.0	HUST_MCLAB [7]. [14]	2017	70.6	—	—	67.9	—	—	—
TextSnake [33]	2018	82.6	1.1	TextProposals +DictNet [6] [37]	2017	56.0	52.3	49.7	53.3	49.6	47.2	0.2
SPCNet [34]	2018	87.2	—	Deep TextSpotter [18]	2017	58.0	53.0	51.0	54.0	51.0	47.0	**9.0**
Mask TextSpotter(1000) [20]	2018	86.0	4.8	Mask TextSpotter(1600) [20]	2018	79.3	74.5	64.2	79.3	73	62.4	2.6
TextNet [21]	2018	87.4	—	TextNet [21]	2018	82.4	78.4	62.4	78.7	74.9	60.5	—
FOTS [22]	2018	88.0	**7.5**	FOTS [22]	2018	84.7	79.3	63.3	81.1	75.9	60.8	7.5
FOTS_MS [22]	2018	89.8	—	FOTS_MS [22]	2018	87.0	82.4	68.0	83.6	79.1	65.3	—
ASTS_Baseline	—	87.8	6.6	ASTS_Baseline	—	85.6	80.5	66.6	82.6	77.0	64.1	6.6
ASTS_Weakly	—	**89.9**	6.6	ASTS_Weakly	—	**87.7**	**82.9**	**68.9**	**84.8**	**79.8**	**66.5**	6.6

'S', 'W' and 'G' represent recognition with strong, weak and generic lexicon respectively.

(1600)" is inference with a scale of 1600. ASTS_Baseline and ASTS_Weakly also achieve very competitive performance both in terms of text detection and recognition. Even though the inference is with a single scale of 1000, ASTS_Weakly outperforms previous multi-scale inference methods. Moreover, as shown by the results for ICDAR 2013 and ICDAR 2015, the methods using a unified framework (such as Mask TextSpotter, FOTS, and our own) perform better than the stepwise methods, since the unified frameworks take full advantage of the correlation and complementarity between text detection and recognition.

Some example results for ICDAR 2015 are presented in Fig. 8. The proposed method overcomes disturbances such as oriented layout (a)(c), small sizes (b), and blur (d). In some extreme cases, such as texts with tiny size (f)(g) or in unusual fonts (h), our proposed method fails to detect or recognize text. In addition, since the attention-based OCR module with an implicit language model is trained using word-level annotations and lacks contextual information, it also probably leads to some recognition failure; for example, in case (f), "WAKA" is recognized as "WAKE".

F. Results for Total-Text

We next evaluated the effectiveness of our method for detecting and recognizing text of arbitrary shape on Total-Text, in which horizontal, oriented and curved text exist simultaneously in most images. Following Mask TextSpotter, the evaluation protocol for both detection and recognition were similar to ICDAR 2015, while the annotation was modified from quadrangles to polygons. The results of our method and comparison with previous works are presented in Table IV. Compared with ICDAR 2013 and ICDAR 2015, both ASTS_Baseline and ASTS_Weakly are far superior to previous methods in terms of both detection and recognition, since most previous methods are not applicable to text with arbitrary shapes. Compared with Mask TextSpotter, which was also developed for detecting and recognizing text with arbitrary shapes, ASTS_Weakly improves detection by 24.2% and end-to-end recognition by over 10%. This significant improvement mainly arises from our framework, which analyzes text from three levels of semantics and implements recognition from both the one- and two-dimensional perspectives, while Mask TextSpotter only acts from the two-dimensional perspective. Some very small texts or serious blurring will probably result in recognition failure if recognition is only from the two-dimensional perspective. However, these cases can be recognized correctly by our OCR module, which implements recognition from the one-dimensional perspective and is relatively stable to font size and text quality. Some example results for Total-Text are presented in Fig. 9. Most texts, even these with various fonts (a)(d)(f) or under uneven light (e), are detected and recognized correctly. However, the excessively small word spacing (g) and rare fonts (h) also probably lead to some failures.

Therefore, according to these experimental results for ICDAR 2013, ICDAR 2015 and Total-Text, our method achieves state-of-the-art performance with respect to both text detection and recognition, especially with irregular text. Additionally, as mentioned above, ICDAR 2015 and Total-Text lack character-level annotations, and characters in these two datasets are equivalent to false-negative samples and would contaminate the training set if we use existing annotations to train our framework. Through weakly supervised learning, we can identify these characters and eliminate this noise of false-negative samples. Therefore, ASTS_Weakly performs better than ASTS_Baseline on all datasets, which also demonstrates that our proposed framework could train a superior model by learning from the weakly annotated datasets.

G. Analysis

1) Discussion of Different Recognition Results: The visualization results revealed, one problem of our method: the performance of the text recognition branch on curved text is unsatisfactory. Fig. 10 (a) depicts an example result of the text recognition branch. Recognition failure occurs because the OCR module in the recognition branch treats text recognition as a sequence-to-sequence task and directly converts the two-dimensional feature maps of curved text into a one-dimensional feature sequence, which loses some useful information and introduces extra noise. Fortunately, these curved texts can be correctly recognized using character instances generated by the detection and mask branch and implemented

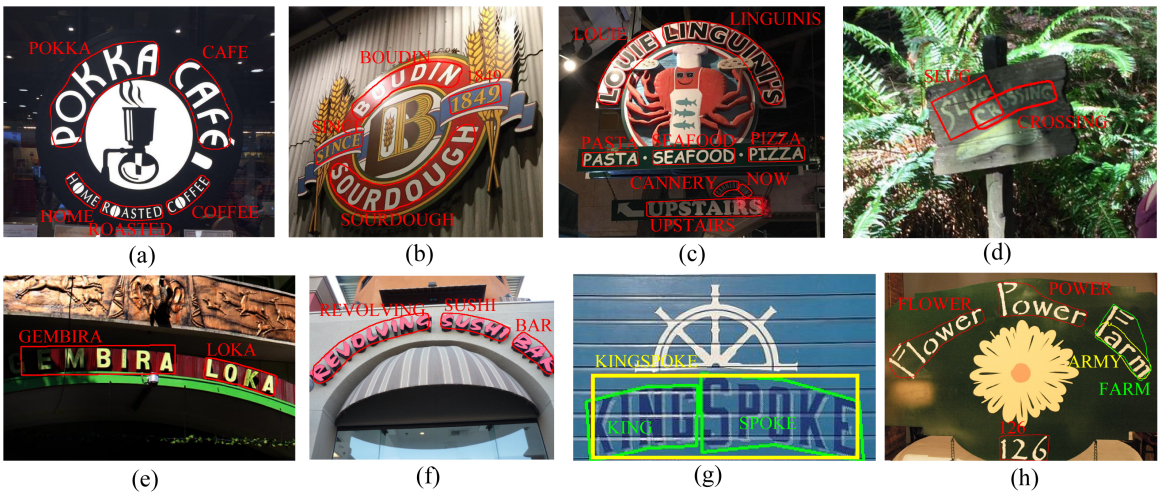


Fig. 9. Example results for Total-Text. The texts and boxes in red represent texts and regions correctly spotted by ASTS. The texts and boxes in yellow represent texts and regions incorrectly spotted by ASTS. The texts and boxes in green represent the texts and regions of the ground truth. Images from (a) to (f) are some success example results, while images from (g) to (h) are failure example results.

TABLE IV

DETECTION AND RECOGNITION RESULTS FOR TOTAL-TEXT

Method	Year	Detection	End-to-End		FPS
			None	Full	
Total-Text [48]	2017	36.0	—	—	—
PixelLink [32]	218	53.1	—	—	—
TextSnake [33]	2018	64.6	—	—	—
SPCNet [34]	2019	82.9	—	—	—
TextBoxes [4]	2017	52.5	36.3	48.9	—
TextNet [21]	2018	63.5	54.0	—	—
Mask TextSpotter [20]	2018	61.3	52.9	71.8	—
ASTS_Baseline	—	83.4	60.8	79.6	4.7
ASTS_Weakly	—	**85.5**	**63.1**	**81.9**	4.7

"None" refers to recognition without any lexicon, and "Full" means recognition with a full lexicon that contains all words in the test set.

Fig. 10. Recognition comparison between the results from a one-dimensional perspective (a) and two-dimensional perspective (b). The polygons in red and green represent word and character instances, respectively.

in the two-dimensional feature space, as shown in Fig. 10 (b). Therefore, our method is capable of recognizing text from the one- and two-dimensional perspectives simultaneously, and the two recognition results can complement each other and improve the final recognition performance.

2) Speed and Model Size: As shown in Tables II, III and IV, when the shorter side of the input image is scaled to 1000, our proposed framework runs at 4.5, 6.6 and 4.7 FPS on ICDAR 2013, ICDAR 2015 and Total-Text, respectively. Although slower than the fastest method [22], it exhibits a good speed-accuracy trade-off. The total number of parameters of our model is 54.3 M, including the backbone network of FPN with ResNet-50.

TABLE V

PERFORMANCE COMPARISONS ON TOTAL-TEXT WITH DIFFERENT BACKBONE NETWORKS

Method	Detection	End-to-End		FPS
		None	Full	
ASTS(ResNet-50)	85.5	63.1	81.9	4.7
ASTS(ResNet-101)	87.3	65.3	84.2	2.5

3) Influence of the Backbone Network: Deeper neural networks have been proven to boost the performance of object detection. To better analyze the capability of our proposed method, we also adopt ResNet-101 as our backbone network and test on Total-Text. As shown in Table V, under the same setting, a depth of the backbone network from 50 to 101 can clearly improve the performance of both text detection and recognition but requires more time cost.

4) Portability to Other Languages: Since our framework is able to recognize texts of arbitrary shapes accurately, we can seamlessly apply our method to detect texts of other languages, including non-occidental ones, if sufficient annotated data are provided for training. Although annotations are needed for training, we can easily adopt the approach proposed by Gupta *et al.* [40] to generate synthetic datasets for other languages with character- and word-level annotations to facilitate the training of our framework for new languages. For real datasets with word-level annotations, we can also search for character samples under weak supervision.

V. CONCLUSION

In this paper, we propose an end-to-end trainable framework for arbitrary shape text spotting by investigating three levels of semantics, *i.e.*, holistic-, pixel-, and sequence-level semantics. In addition to the shared network, which is used to extract image feature maps and generate region proposals, our

framework has three branches: the text detection branch, the text mask branch, and the text recognition branch. The text detection and mask branches explore holistic- and pixel-level semantics to determine text shape and realize recognition from the two-dimensional perspective. The text recognition branch is implemented from the one-dimensional perspective based on sequence-level semantics and is mainly used to recognize text that failed character detection in the previous branches. Finally, we merge the two recognition results. Our experimental results show that our framework achieves impressive performance in detecting and recognizing text of arbitrary shape. Moreover, our proposed framework can also perform well on weakly supervised data.

REFERENCES

[1] Q. Ye and D. Doermann, "Text detection and recognition in imagery: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 7, pp. 1480–1500, Jul. 2015.

[2] P. He, W. Huang, Y. Qiao, C. C. Loy, and X. Tang, "Reading scene text in deep convolutional sequences," in *Proc. 30th AAAI Conf. Artif. Intell.*, 2016, pp. 1–9.

[3] W. He, X.-Y. Zhang, F. Yin, and C.-L. Liu, "Deep direct regression for multi-oriented scene text detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 745–753.

[4] M. Liao, B. Shi, X. Bai, X. Wang, and W. Liu, "Textboxes: A fast text detector with a single deep neural network," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 4161–4167.

[5] 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.

[6] L. Gómez and D. Karatzas, "TextProposals: A text-specific selective search algorithm for word spotting in the wild," *Pattern Recognit.*, vol. 70, pp. 60–74, Oct. 2017.

[7] B. Shi, X. Bai, and S. Belongie, "Detecting oriented text in natural images by linking segments," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2550–2558.

[8] 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. (ECCV)*, 2016, pp. 56–72.

[9] X. Zhou *et al.*, "EAST: An efficient and accurate scene text detector," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5551–5560.

[10] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman, "Reading text in the wild with convolutional neural networks," *Int. J. Comput. Vis.*, vol. 116, no. 1, pp. 1–20, Jan. 2016.

[11] 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. (CVPR)*, Jun. 2016, pp. 4168–4176.

[12] 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.

[13] L. Neumann and J. Matas, "Real-time lexicon-free scene text localization and recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 9, pp. 1872–1885, Sep. 2016.

[14] 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.

[15] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.

[16] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[17] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2961–2969.

[18] 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. (ICCV)*, Oct. 2017, pp. 2204–2212.

[19] 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. (ICCV)*, Oct. 2017, pp. 5238–5246.

[20] 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. (ECCV)*, 2018, pp. 67–83.

[21] 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. (ACCV)*, 2018, pp. 83–99.

[22] 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/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5676–5685.

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

[24] 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.

[25] 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.*, Dec. 2013, pp. 1241–1248.

[26] L. Neumann and J. Matas, "Scene text localization and recognition with oriented stroke detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 97–104.

[27] M. Buta, L. Neumann, and J. Matas, "FASText: Efficient unconstrained scene text detector," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1206–1214.

[28] K. In Kim, K. Jung, and J. Hyung Kim, "Texture-based approach for text detection in images using support vector machines and continuously adaptive mean shift algorithm," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 12, pp. 1631–1639, Dec. 2003.

[29] X. Chen and A. L. Yuille, "Detecting and reading text in natural scenes," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 2, Jun. 2004, p. 2.

[30] S. Zhu and R. Zanibbi, "A text detection system for natural scenes with convolutional feature learning and cascaded classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 625–632.

[31] W. Liu *et al.*, "SSD: Single shot MultiBox detector," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2016, pp. 21–37.

[32] D. Deng, H. Liu, X. Li, and D. Cai, "Pixellink: Detecting scene text via instance segmentation," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 6773–6780.

[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 *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 20–36.

[34] E. Xie, Y. Zang, S. Shao, G. Yu, C. Yao, and G. Li, "Scene text detection with supervised pyramid context network," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, Jul. 2019, pp. 9038–9045.

[35] M. Jaderberg, A. Vedaldi, and A. Zisserman, "Deep features for text spotting," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2014, pp. 512–528.

[36] A. Bissacco, M. Cummins, Y. Netzer, and H. Neven, "PhotoOCR: Reading text in uncontrolled conditions," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 785–792.

[37] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman, "Synthetic data and artificial neural networks for natural scene text recognition," 2014, *arXiv:1406.2227*. [Online]. Available: http://arxiv.org/abs/1406.2227

[38] C.-Y. Lee and S. Osindero, "Recursive recurrent nets with attention modeling for OCR in the wild," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2231–2239.

[39] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.

[40] A. Gupta, A. Vedaldi, and A. Zisserman, "Synthetic data for text localisation in natural images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2315–2324.

[41] Z. Chen, S. Huang, and D. Tao, "Context refinement for object detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 71–86.

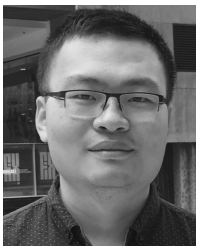
[42] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2117–2125.

[43] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[44] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 577–585.

[45] Z.-H. Zhou, "A brief introduction to weakly supervised learning," *Nat. Sci. Rev.*, vol. 5, no. 1, pp. 44–53, 2017.
[46] D. Karatzas *et al.*, "ICDAR 2013 robust reading competition," in *Proc. 12th Int. Conf. Document Anal. Recognit.*, Aug. 2013, pp. 1484–1493.
[47] D. Karatzas *et al.*, "ICDAR 2015 competition on robust reading," in *Proc. 13th Int. Conf. Document Anal. Recognit. (ICDAR)*, Aug. 2015, pp. 1156–1160.
[48] C. K. Ch'ng and C. S. Chan, "Total-text: A comprehensive dataset for scene text detection and recognition," in *Proc. 14th IAPR Int. Conf. Document Anal. Recognit. (ICDAR)*, vol. 1, Nov. 2017, pp. 935–942.
[49] S. Zhang, M. Lin, T. Chen, L. Jin, and L. Lin, "Character proposal network for robust text extraction," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2016, pp. 2633–2637.
[50] F. Massa and R. Girshick. (2018). *MaskrCNN-Benchmark: Fast, Modular Reference Implementation of Instance Segmentation and Object Detection Algorithms in PyTorch*. Accessed: Sep. 1, 2019. [Online]. Available: https://github.com/facebookresearch/maskrcnn-benchmark

Juhua Liu received the M.S. and Ph.D. degrees in graphic communication from the School of Printing and Packaging, Wuhan University, Wuhan, China, in 2007 and 2014, respectively. He is currently an Associate Professor with the School of Printing and Packaging, Wuhan University. His research interests mainly include image processing, computer vision and machine learning.



Zhe Chen received the B.S. degree in computer science from the University of Science and Technology of China, Hefei, China, in 2014, the Ph.D. degree from UBTECH Sydney Artificial Intelligence Centre, School of Computer Science, Faculty of Engineering and Information Technologies, The University of Sydney, in 2019. His research interests include object detection, computer vision, and deep learning. His studies have been published in the IEEE COMPUTER VISION AND PATTERN RECOGNITION, ICONIP, ECCV, and JAS. He also serves as a reviewer for a number of journals and conferences such as TIP, TCSVT, T-CYB, and so on.

Bo Du (Senior Member, IEEE) received the Ph.D. degree in photogrammetry and remote sensing from the State Key Laboratory of Information Engineering in Surveying, Mapping and Remote sensing, Wuhan University, Wuhan, China, in 2010. He is currently a Professor with the National Engineering Research Center for Multimedia Software, Institute of Artificial Intelligence, and School of Computer Science, Wuhan University. He has published more than 60 research articles in the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEM (TNNLS), the IEEE TRANSACTIONS ON IMAGE PROCESSING (TIP), the IEEE TRANSACTIONS ON MULTIMEDIA (TMM), the IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING (TGRS), the IEEE JOURNAL OF SELECTED TOPICS IN EARTH OBSERVATIONS AND APPLIED REMOTE SENSING (JSTARS), and the IEEE GEOSCIENCE AND REMOTE SENSING LETTERS (GRSL). Thirteen have been ESI hot articles or highly cited articles. His major research interests include pattern recognition, hyperspectral image processing, machine learning, and signal processing. He is currently a Senior Member of the IEEE. He received a Highly Cited Researcher 2019 Award from the Web of Science Group, the Distinguished Paper Award from IJCAI 2018, the Best Paper Award of the IEEE Whispers 2018 and the Champion Award of the IEEE Data Fusion Contest 2018. He received the Best Reviewer Award from IEEE GRSS for his service to IEEE Journal of Selected Topics in Earth Observations and Applied Remote Sensing (JSTARS) in 2011 and an ACM Rising Star Award for his academic progress in 2015. He serves as associate editor for Pattern Recognition and Neurocomputing, senior PC for IJCAI/AAAI/KDD, and area chair for ICPR and IJCNN. He was the Session Chair for both the International Geoscience and Remote Sensing Symposium (IGARSS) 2018/2016 and the 4th IEEE GRSS Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS). He also serves as a reviewer of 20 Science Citation Index (SCI) journals, including IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING, TIP, JSTARS, and GRSL.

Dacheng Tao (Fellow, IEEE) is currently a Professor of computer science and ARC Laureate Fellow with the School of Computer Science and the Faculty of Engineering, and the Inaugural Director of the UBTECH Sydney Artificial Intelligence Centre, at The University of Sydney. His research results in artificial intelligence have expounded in one monograph and 200+ publications at prestigious journals and prominent conferences, such as IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, IJCV, JMLR, AIJ, AAAI, IJCAI, NIPS, ICML, CVPR, ICCV, ECCV, ICDM, and KDD, with several Best Paper Awards. He received the 2018 IEEE International Conference on Data Mining Research Contributions Award and the 2015 Australian Scopus-Eureka prize. He is a Fellow of AAAS, ACM, and Australian Academy of Science.