Character Region Attention For Text Spotting

Youngmin Baek, Seung Shin, Jeonghun Baek, Sungrae Park, Junyeop Lee, Daehyun Nam, and Hwalsuk Lee*

Clova AI Research, NAVER Corp.
{youngmin.baek,seung.shin,jh.baek,sungrae.park,junyeop.lee, daehyun.nam,hwalsuk.lee}@navercorp.com

Abstract. A scene text spotter is composed of text detection and recognition modules. Many studies have been conducted to unify these modules into an end-to-end trainable model to achieve better performance. A typical architecture places detection and recognition modules into separate branches, and a RoI pooling is commonly used to let the branches share a visual feature. However, there still exists a chance of establishing a more complimentary connection between the modules when adopting recognizer that uses attention-based decoder and detector that represents spatial information of the character regions. This is possible since the two modules share a common sub-task which is to find the location of the character regions. Based on the insight, we construct a tightly coupled single pipeline model. This architecture is formed by utilizing detection outputs in the recognizer and propagating the recognition loss through the detection stage. The use of character score map helps the recognizer attend better to the character center points, and the recognition loss propagation to the detector module enhances the localization of the character regions. Also, a strengthened sharing stage allows feature rectification and boundary localization of arbitrary-shaped text regions. Extensive experiments demonstrate state-of-the-art performance in publicly available straight and curved benchmark dataset.

Keywords: Optical character recognition (OCR), Character region attention, Text spotting, Scene text detection, Scene text recognition

1 Introduction

Scene text spotting, including text detection and recognition, has recently attracted much attention because of its variety of applications in instant translation, image retrieval, and scene parsing. Although existing text detectors and recognizers work well on horizontal texts, it still remains as a challenge when it comes to spotting curved text instances in scene images.

To spot curved texts in an image, a classic method is to cascade existing detection and recognition models to manage text instances on each side. The detectors[32, 31, 2] attempt to capture the geometric attributes of curved texts

* Corresponding author.

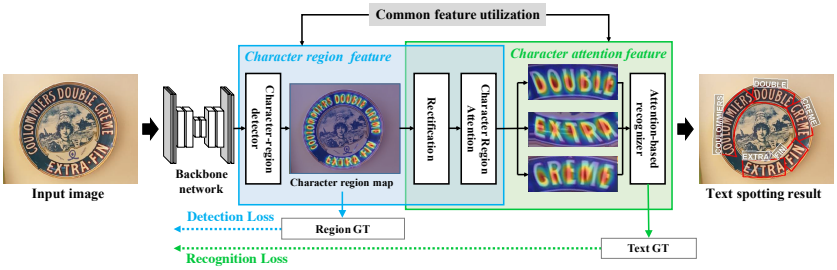

Fig. 1. Concept of the proposed method. The character region feature from the detector is used as an input character attention feature to the recognizer. Having a tightly coupled architecture lets the recognition loss flow through the whole network.

by applying complicated post-processing techniques, and the recognizers apply multi-directional encoding[6] or take rectification modules[37, 46, 11] to enhance the accuracy of the recognizer on curved texts.

As deep learning advanced, researches have been made to combine detectors and recognizers into a jointly trainable end-to-end network [14, 29]. Having a unified model not only provides efficiency in the size and speed of the model, but also helps the model learn a shared feature that pulls up the overall performance. To gain benefit from this property, attempts have also been made to handle curved text instances using an end-to-end model[32, 34, 10, 44]. However, most of the existing works only adopt a RoI pooling to share low-level feature layers between detection and recognition branches. In the training phase, instead of training the whole network, only shared feature layers are trained using both detection and recognition losses.

As shown in Fig. 1, we propose a novel end-to-end *Character Region Attention For Text Spotting* model, referred to as CRAFTS. Instead of isolating detection and recognition modules in two separate branches, we form a single pipeline by establishing a complimentary connection between the modules. We observe that recognizer [1] using attention-based decoder and detector [2] encapsulating character spatial information share a common sub-task which is to localize character regions. By tightly integrating the two modules, the outputs from the detection stage helps recognizer attend better to the character center points, and loss propagated from recognizer to detector stage enhances the localization of the character regions. Furthermore, the network is able to maximize the quality of the feature representation used in the common sub-tasks. To best of our knowledge, this is the first end-to-end work that constructs a tightly coupled loss propagation.

The summary of our contribution follows; (1) We propose an end-to-end network that could detect and recognize arbitrary-shaped texts. (2) We construct a complementary relationship between the modules by utilizing spatial character information from the detector on the rectification and recognition module. (3) We establish a single pipeline by propagating the recognition loss throughout all the features in the network. (4) We achieve the state-of-the-art performances in IC13, IC15, IC19-MLT, and TotalText [20, 19, 33, 7] datasets that contain numerous horizontal, curved and multilingual texts.

2 Related Work

Text detection and recognition methods Detection networks use regression based [16, 24, 25, 48] or segmentation based [9, 31, 43, 45] methods to produce text bounding boxes. Some recent methods like [17, 26, 47] take Mask-RCNN [13] as the base network and gain advantages from both regression and segmentation methods by employing multi-task learning. In terms of units for text detection, all methods could also be sub-categorized depending on the use of word-level or character-level[16, 2] predictions.

Text recognizers typically adopt CNN-based feature extractor and RNN based sequence generator, and are categorized by their sequence generators; connectionist temporal classification (CTC) [35] and attention-based sequential decoder [21, 36]. Detection model provides information of the text regions, but it is still a challenge for the recognizer to extract useful information in arbitrary-shaped texts. To help recognition networks handle irregular texts, some researches [36, 28, 37] utilize spatial transformer network (STN)[18]. Also, the papers [11, 46] further extend the use of STN by iterative executing the rectification method. These studies show that running STN recursively helps recognizer extract useful features in extremely curved texts. In [27], Recurrent RoIWarp Layer was proposed to crop individual characters before recognizing them. The work proves that the task of finding a character region is closely related to the attention mechanism used in the attention-based decoder.

One way to construct a text spotting model is to sequentially place detection and recognition networks. A well known two-staged architecture couples TextBox++[24] detector and CRNN[35] recognizer. With its simplicity, the method achieves favorable results.

End-to-end using RNN-based recognizer EAA[14] and FOTS[29] are end-to-end models based on EAST detector [49]. The difference between these two networks lies in the recognizer. The FOTS model uses CTC decoder [35], and the EAA model uses attention decoder [36]. Both works implement an affine transformation layer to pool the shared feature. The proposed affine transformation works well on horizontal texts, but shows limitations when handling arbitrary-shaped texts. TextNet [42] proposed a spatial-aware text recognizer with perspective-RoI transformation in the feature pooling layer. The network keeps an RNN layer to recognize a sequence of text in the 2D feature map, but due to the lack of expressively of the quadrangles, the network still shows limitations when detecting curved texts.

Qin et al. [34] proposed a Mask-RCNN[13] based end-to-end network. Given the box proposals, features are pooled from the shared layer and the ROI-masking layer is used to filter out the background clutters. The proposed method increases its performance by ensuring attention only in the text region. Busta et al. proposed Deep TextSpotter [3] network and extended their work in E2E-MLT [4]. The network is composed of FPN based detector and a CTC-based recognizer. The model predicts multiple languages in an end-to-end manner.

End-to-end using CNN-based recognizer Most CNN-based models that recognize texts in character level have advantages when handling arbitrary-shaped texts. MaskTextSpotter [32] is a model that recognizes text using a segmentation approach. Although it has strengths in detecting and recognizing individual characters, it is difficult to train the network since character-level annotations are usually not provided in the public datasets. CharNet [44] is another segmentation-based method that makes character level predictions. The model is trained in a weakly-supervised manner to overcome the lack of character-level annotations. During training, the method performs iterative character detection to create pseudo-ground-truths.

While segmentation-based recognizers have shown great success, the method suffers when the number of target characters increases. Segmentation based models require more output channels as the number of character sets grow, and this increases memory requirements. The journal version of MaskTextSpotter[23] expands the character set to handle multiple languages, but the authors added a RNN-based decoder instead of using their initially proposed CNN-based recognizer. Another limitation of segmentation-based recognizer is the lack of contextual information in the recognition branch. Due to the absence of sequential modeling like RNNs, the accuracy of the model drops under noisy images.

TextDragon [10] is another segmentation-based method that localize and recognize text instances. However, a predicted character segment is not guaranteed to cover a single character region. To solve the issue, the model incorporates CTC to remove overlapping characters. The network shows good detection performance but shows limitations in the recognizer due to the lack of sequential modeling.

3 Methodology

3.1 Overview

Proposed CRAFTS network can be divided into three stages; detection stage, sharing stage, and recognition stage. A detailed pipeline of the network is illustrated in Fig. 2. Detection stage takes an input image and localizes oriented text boxes. Sharing stage then pools backbone high-level features and detector outputs. The pooled features are then rectified using the rectification module, and are concatenated together to form a *character attended feature*. In the recognition stage, attention-based decoder predicts text labels using the *character attended feature*. Finally, a simple post-processing technique is optionally used for better visualization.

3.2 Detection Stage

CRAFT detector[2] is selected as a base network because of its capability of representing semantic information of the character regions. The outputs of the CRAFT network represent center probability of character regions and linkage

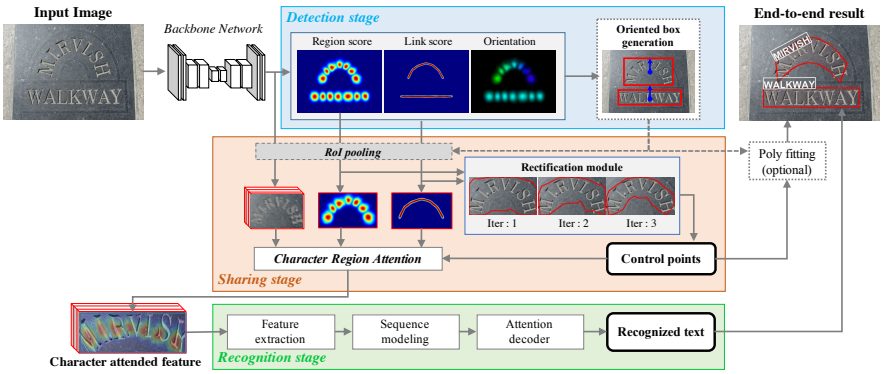

Fig. 2. Schematic overview of CRAFTS pipeline.

between them. We contemplate that this character centeredness information can be used to support the attention module in the recognizer since both modules aim to localize the center position of characters. In this work, we make three changes in the original CRAFT model; backbone replacement, link representation, and orientation estimation.

Backbone replacement Recent studies show that the use of ResNet50 captures well-defined feature representations of both the detector and the recognizer[30, 1]. We therefore replace the backbone of the network from VGG-16[40] to ResNet50[15].

Link representation The occurrence of vertical texts is not common in Latin texts, but it is frequently found in East Asian languages like Chinese, Japanese, and Korean. In this work, a binary center line is used to connect the sequential character regions. This change was made because employing the original affinity maps on vertical texts often produced ill-posed perspective transformation that generated invalid box coordinates. To generate ground truth linkmap, a line segment with thickness t is drawn between adjacent characters. Here, $t = max((d_1 + d_2)/2 * \alpha, 1)$, where d_1 and d_2 are the diagonal lengths of adjacent character boxes and α is the scaling coefficient. Use of the equation lets the width of the center line proportional to the size of the characters. We set α as 0.1 in our implementation.

Orientation estimation It is important to obtain the right orientation of text boxes since the recognition stage requires well-defined box coordinates to recognize the text properly. To this end, we add two-channel outputs in the detection stage; channel is used to predict angles of characters along the x-axis, y-axis each. To generate the ground truth of orientation map, the upward angle of the GT character bounding box is represented as θ^*_{box}, the channel predicting x-axis has a value of $S^*_{cos}(p) = (\cos\theta + 1) \times 0.5$, and the channel predicting y-axis has a value of $S^*_{sin}(p) = (\sin\theta + 1) \times 0.5$. The ground truth orientation map is generated by filling the pixels p in the region of the word box with the values of $S^*_{cos}(p)$ and $S^*_{sin}(p)$. The trigonometric function is not directly used to let the channels have the same output range with the region map and the link map; between 0 and 1.

The loss function for orientation map is calculated by Eq. 1.

$$L_\theta = S_r^*(p) \cdot (||S_{sin}(p) - S_{sin}^*(p)||_2^2 + ||S_{cos}(p) - S_{cos}^*(p)||_2^2) \tag{1}$$

where $S_{sin}^*(p)$ and $S_{cos}^*(p)$ denote the ground truth of text orientation. Here, the character region score $S_r(p)$ is used as a weighting factor because it represents the confidence of the character centeredness. By doing this, the orientation loss is calculated only in the positive character regions.

The final objective function in the detection stage L_{det} is defined as,

$$L_{det} = L_r + L_l + \lambda L_\theta \tag{2}$$

where L_r and L_l denote character region loss and link loss, which are exactly same in [2]. The L_θ is the orientation loss, and is multiplied with λ to control the weight. In our experiment, we set λ to 0.1.

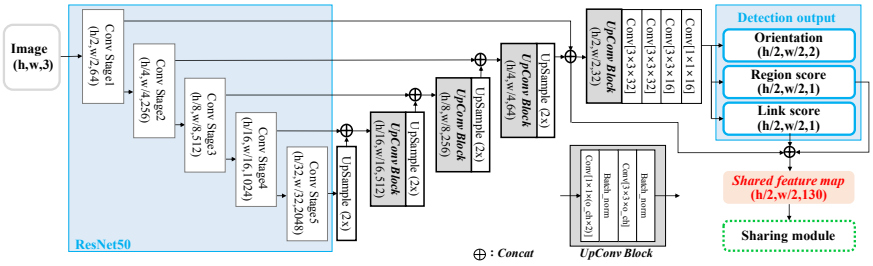

Fig. 3. Schematic illustration of the backbone network and the detection head.

The architecture of the backbone and modified detection head is illustrated in Fig. 3. The final output of the detector has four channels, each representing *character region map* S_r, *character link map* S_l, and two *orientation maps* S_{sin}, S_{cos}.

During inference, we apply the same post-processing as described in [2] to obtain text bounding boxes. First, by using predefined threshold values, we make binary maps of *character region map* S_r and *character link map* S_l. Then, using the two maps, the text blobs are constructed by using connected components labeling(CCL). The final boxes are obtained by finding a minimum bounding box enclosing each text blob. We additionally determine the orientation of the bounding box by utilizing pixel-wise averaging scheme. As shown in the Eq. 3, the angle of the text box is found by taking the arctangent of accumulated sine and cosine values at the predicted orientation map.

$$\theta_{box} = \arctan\left(\frac{\sum(S_r(p) \times (S_{sin}(p) - 0.5)}{\sum(S_r(p) \times (S_{cos}(p) - 0.5)}\right) \tag{3}$$

θ_{box} denotes orientation of the text box, S_{cos} and S_{sin} are the 2-ch orientation outputs. The same character centerdeness-based weighting scheme that used in the loss calculation is applied to predict the orientation as well.

3.3 Sharing Stage

Sharing stage consists of two modules: text rectification module and character region attention(CRA) modules. To rectify arbitrarily-shaped text region, a thin-plate spline (TPS) [37] transformation is used. Inspired by the work of [46], our rectification module incorporates iterative-TPS to acquire a better representation of the text region. By updating the control points attractively, the curved geometry of a text in an image becomes ameliorated. Through empirical studies, we discover that three TPS iterations are sufficient for rectification.

Typical TPS module takes an word image as input, but we feed the character region map and link map since they encapsulate geometric information of the text regions. We use twenty control points to tightly cover the curved text region. To use these control points as a detection result, they are transformed to the original input image coordinate. We optionally perform 2D polynomial fitting to smooth the bounding polygon. Examples of iterative-TPS and final smoothed polygon output are shown in Fig. 4.

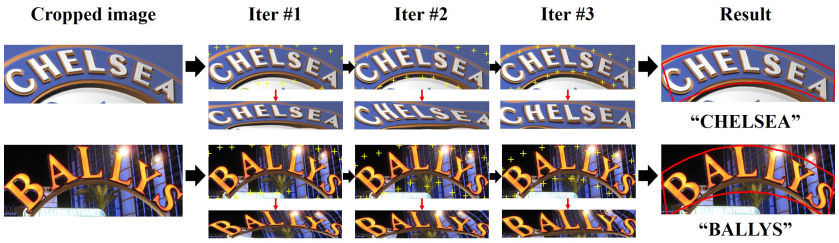

Fig. 4. Example of iterative TPS. The middle rows show TPS control points on each iteration, and the bottom row shows rectified images on each stage. The control points are drawn in the image level for better visualization. Actual rectification is done in the feature space. Final result was smoothed using a 2D polynomial.

CRA module is the key component that tightly couples detection and recognition modules. By simply concatenating rectified character score map with feature representation, the model establishes following advantages. Creating a link between detector and recognizer allows recognition loss to propagate through detection stage, and this improves the quality of character score map. Also, attaching character region map to the feature helps recognizer attend better to the character regions. Ablation study of using this module will be discussed further in the experiment section.

3.4 Recognition Stage

The modules in the recognition stage are formed based on the results reported in [1]. There are three components in the recognition stage: feature extraction, sequence modeling, and prediction. The feature extraction module is made lighter than a solitary recognizer since it takes high-level semantic features as input.

Detailed architecture of the module is shown in Table 1. After extracting the features, a bidirectional LSTM is applied for sequence modeling, and attention-based decoder makes a final text prediction.

Layers	Configurations	Output
Input	pooled feature	$64 \times 16 \times 130$
Block1	$\begin{bmatrix} \text{c}:256, \text{k}:3 \times 3 \\ \text{c}:256, \text{k}:3 \times 3 \end{bmatrix} \times 2$	$64 \times 16 \times 256$
Conv1	c: 256 k: 3×3	$64 \times 16 \times 256$
MaxPool	k: 2×2 s: 1×2 p: 1×0	$65 \times 8 \times 256$
Block2	$\begin{bmatrix} \text{c}:512, \text{k}:3 \times 3 \\ \text{c}:256, \text{k}:3 \times 3 \end{bmatrix} \times 5$	$65 \times 8 \times 512$
Conv2	c: 512 k: 3×3	$65 \times 8 \times 512$
Block3	$\begin{bmatrix} \text{c}:512, \text{k}:3 \times 3 \\ \text{c}:512, \text{k}:3 \times 3 \end{bmatrix} \times 3$	$65 \times 8 \times 512$
Conv3	c: 512 k: 2×2 s: 1×2 p: 1×0	$65 \times 4 \times 512$
Conv4	c: 512 k: 2×2 s: 1×1 p: 0×0	$65 \times 3 \times 512$
AvgPool	k: 1×3 s: 1×2 p: 1×0	$65 \times 1 \times 512$

Table 1. A simplified ResNet feature extraction module .

At each time step, attention-based recognizer decodes textual information by masking attention outputs to the features. Although attention module works well in most cases, it fails to predict characters when attention points are misaligned or vanished [5, 14]. Fig. 5 shows the effect of using CRA module. Well-placed attention points allow robust text prediction.

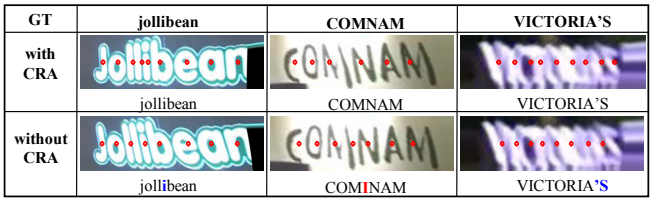

Fig. 5. Attention problems with and without Character Region Attention module. The red dots represent attention points of the decoding characters. Missing characters are colored in blue, and misrecognized characters are colored in red. The cropped images are slightly different since generated control points in each rectification modules are inconsistent.

The objective function, L_{reg}, in the recognition stage is

$$L_{reg} = -\sum_{i} \log p(Y_i|X_i) \tag{4}$$

where $p(Y_i|X_i)$ indicates the generation probability of the character sequence, Y_i, from the cropped feature representation, X_i of the i-th word box.

The final loss, L, used for training is composed of detection loss and recognition loss by taking $L = L_{det} + L_{reg}$. The overall flow of the recognition loss is shown in Fig. 6. The loss flows through the weights in the recognition stage, and propagates towards detection stage through *Character Region Attention* module. Detection loss on the other hand is used as an intermediate loss, and thus the weights before detection stage are updated using both detection and recognition losses.

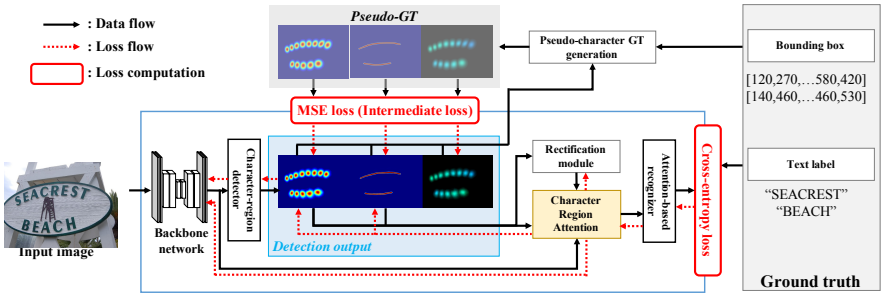

Fig. 6. The entire loss flow of CRAFTS model.

4 Experiment

4.1 Datasets

English datasets *IC13* [20] dataset consists of high-resolution images, 229 for training and 233 for testing. A rectangular box is used to annotate word-level text instances. *IC15* [20] consists of 1000 training and 500 testing images. A quadrilateral box is used to annotate word-level text instances. *TotalText* [7] has 1255 training and 300 testing images. Unlike IC13 and IC15 datasets, it contains curved text instances and is annotated using polygon points.
Multi-language dataset *IC19* [33] dataset contains 10,000 training and 10,000 testing images. The dataset contains texts in 7 different languages and is annotated using quadrilateral points.

4.2 Training strategy

We jointly train both the detector and recognizer in the CRAFTS model. To train the detection stage, we follow the weakly-supervised training method described in [2]. The recognition loss is calculated by making a batch of randomly sampled cropped word features in each image. Maximum number of words per image is set to 16 to prevent out-of-memory error. Data augmentations in the detector apply techniques like crops, rotations, and color variations. For the recognizer, the corner points of the ground truth boxes are perturbed in a range between 0 to 10% of the shorter length of the box.

The model is first trained on the SynthText dataset [12] for 50k iterations, and we further train the network on target datasets. Adam optimizer is used, and *On-line Hard Negative Mining(OHEM)* [39] is applied to enforce 1:3 ratio of

Method	IC13(Det)			IC13(E2E)			IC15(Det)			IC15(E2E)			FPS
	R	P	H	S	W	G	R	P	H	S	W	G	
Deep TextSpotter[3]	-	-	-	89	86	77	-	-	-	54	51	47	9
TextBoxes++*[24]	86	92	89	93	92	85	78.5	87.8	82.9	73.3	65.8	51.9	-
TextNet*[42]	89.1	93.6	91.3	89.7	88.8	82.9	80.8	85.7	83.2	78.6	74.9	60.4	2.7
EAA[14]	89	91	90	91	89	86	86	87	87	82	77	63	-
TextDragon[10]	-	-	-	-	-	-	83.7	92.4	87.8	82.5	78.3	65.1	-
FOTS*[29]	-	-	92.8	91.9	90.1	84.7	87.9	91.8	89.8	83.5	79.1	65.3	7.5
Li et al.[22]	80.5	91.4	85.6	92.5	91.2	84.9	-	-	-	84.4	78.9	66.1	1.3
Qin et al.[34]	-	-	-	-	-	-	87.9	91.6	89.7	**85.5**	81.9	69.9	4.7
CharNet*[44]	-	-	-	-	-	-	**90.4**	**92.6**	**91.5**	85.0	81.2	71.0	-
MaskTextSpotter*[23]	89.5	94.8	92.1	93.3	91.3	88.2	87.3	86.6	87.0	83.0	77.7	73.5	2.0
CRAFTS(ours)	**90.9**	**96.1**	**93.4**	**94.2**	**93.8**	**92.2**	85.3	89.0	87.1	83.1	**82.1**	**74.9**	5.4

Table 2. Results on horizontal Latin datasets. * denote the results based on multi-scale tests. R, P, and H refer to recall, precision and H-mean, and S, W, and G indicate strongly-, weakly- and generic-contextualization results, respectively. The best score is highlighted in **bold**. The evaluation metric of ICDAR 2013 detection task is DetEval, and IoU metric is used for other three cases. FPS is for reference only due to the different experimental environments.

positive and negative pixels in the detection loss. When fine-tuning the model, SynthText dataset is mixed with the ratio of 1:5. We take 94 characters to cover alphabets, numbers, and special characters, and take 4267 characters for the multi-language dataset.

4.3 Experimental Results

Horizontal datasets (IC13, IC15) To target the IC13 benchmark, we take the model trained on the SynthText dataset and perform finetuning on IC13 and IC19 datasets. During inference, we resize the longer side of the input to 1280. The results show significant increase in performance when compared with the previous state-of-the-art works.

The model trained on IC13 dataset is then fine-tuned on the IC15 dataset. During the evaluation process, the input size of the model is set to 2560x1440. Note that we perform generic evaluation without the generic vocabulary set. The quantitative results on IC13 and IC15 datasets are listed in Table. 2.

Our method surpasses previous methods in both generic and weakly- contextualization end-to-end tasks, and shows comparable results in other tasks. The generic performance is meaningful because a vocabulary set is not provided in practical scenarios. Note that we get slightly low detection scores on IC15 dataset and also observe low performance in strongly-contextualization results. The relatively low detection performance is obtained mainly due to the granularity difference, and will be discussed further in the later section.

Curved datasets (TotalText) From the model trained on IC13 dataset, we further train the model on TotalText dataset. During inference, we resize the longer side of the input to 1920, and the control points from rectification module are used for detector evaluation. The qualitative results are shown in Fig. 7.

The character region map and the link map are illustrated using a heatmap, and the weighted pixel-wise angle values are visualized in the HSV color space. As it is shown in the figure, the network successfully localizes polygon regions and recognizes characters in the curved text region. Two top-left figures show successful recognition of fully rotated and highly curved text instances.

Method	Detection			E2E(None)			E2E(Full)
	R	P	H	R	P	H	H
TextDragon[10]	75.7	85.6	80.3	-	-	48.4	74.8
TextNet[42]	59.4	68.2	63.5	56.4	51.9	54.0	-
Li et al.[22]	59.8	64.8	62.2	-	-	57.8	-
MaskTextSpotter[23]	75.4	81.8	78.5	-	-	65.3	77.4
CharNet*[44]	85.0	88.0	86.5	-	-	69.2	-
Qin et al.†[34]	85.0	87.8	86.4	-	-	70.7	-
CRAFTS(ours)	**85.4**	**89.5**	**87.4**	**72.2**	**86.5**	**78.7**	-

Table 3. Results on TotalText dataset. None means no lexicon is used for contextualization. The full lexicon contains all words in the test set. * indicates the multi-scale inference, and † denotes models trained on the private datasets.

Quantitative results on TotalText dataset are listed in Table. 3. DetEval[7] evaluates the performance of the detector and modified IC15 evaluation scheme measures the end-to-end performance. Our method outperforms previously reported methods by a large margin. Note that even without the vocabulary set, the end-to-end result significantly exceeds the h-mean score by 8.0%.

Multi-language dataset (IC19) Evaluation on multiple languages is performed using IC19-MLT dataset. The output channel in the prediction layer of the recognizer was expanded to 4267 to handle the characters in Arabic, Latin, Chinese, Japanese, Korean, Bangladesh, and Hindi. However, occurrence of characters in the dataset is not evenly distributed. Among 4267 characters in the training set, 1017 characters occur once in the dataset, and this insufficiency makes it hard for the model to make accurate label predictions. To solve class imbalance problem, we first freeze the weights in the detection stage and pretrain the weights in the recognizer with other publicly available multi-language datasets: SynthMLT, ArT, LSVT, ReCTS and RCTW [4, 8, 41, 38]. We then let the loss flow through the whole network and use IC19 dataset to finetune the model. Since no paper reports performance, we compare our results with E2E-MLT [4, 33]. The samples from the IC19 dataset are shown in Fig. 8. We hope our study is set as a baseline for future works on the IC19-MLT benchmark.

Method	Detection			E2E		
	R	P	H	R	P	H
E2E-MLT[4]	-	-	-	20.5	37.4	26.5
CRAFTS(ours)	**70.1**	**81.7**	**75.5**	**48.5**	**72.9**	**58.2**

Table 4. Results on IC19-MLT dataset.

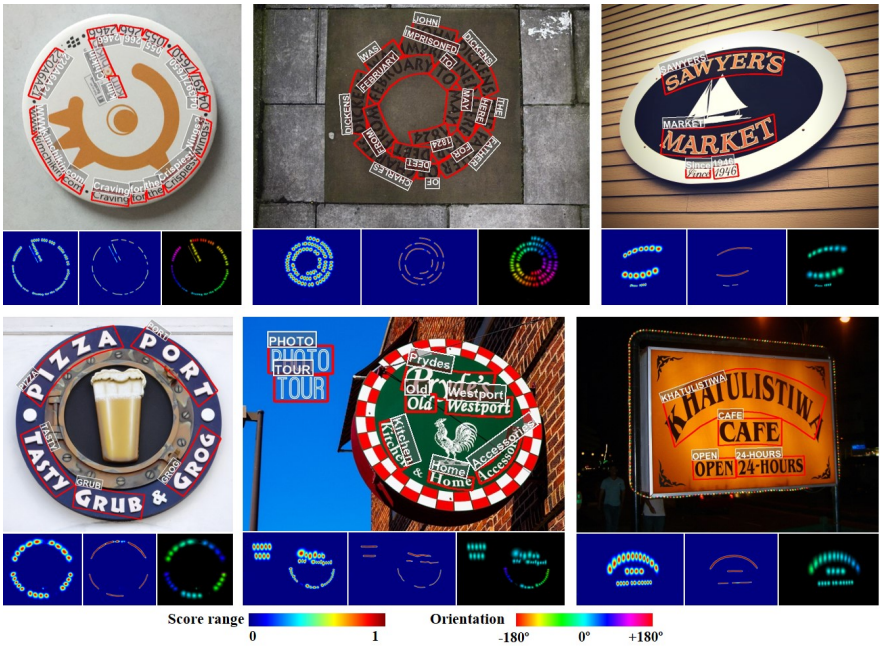

Fig. 7. Results on the TotalText dataset. First row: each column shows the input image (top) with its respective region score map, link map, and orientation map.

4.4 Ablation study

Attention assisted by *Character Region Attention* In this section, we study how *Character Region Attention(CRA)* affects the performance of the recognizer by training a separate network without CRA.

Table. 5 shows the effect of using CRA on benchmark datasets. Without CRA, we observe performance drops on all of the datasets. Especially on the perspective dataset (IC15) and the curved dataset (TotalText), we observe a greater gap when compared with the horizontal dataset (IC13). This implies that feeding character attention information improves the performance of the recognizer when dealing with irregular texts.

Dataset	Type	CRA	R	P	H	Gain
IC13	**Horizontal**	-	89.2	94.8	91.9	-
		✓	89.5	95.0	92.2	+0.3
IC15	**Perspective**	-	64.9	84.1	73.2	-
		✓	65.9	86.7	74.9	+1.7
TotalText	**Curved**	-	71.9	84.0	77.5	-
		✓	72.2	86.5	78.7	+1.2

Table 5. The end-to-end performance comparisons of using character attention maps. CRA denotes the use of *Character Region Attention* in recognition stage. R, P, and H refers to Recall, Precision and Hmean values.

Fig. 8. Qualitative results on IC19 MLT dataset.

Recognition loss in the detection stage The recognition loss flowing through the detection stage affects the quality of the character region map and character link map. It is expected that the recognition loss helps detector localize character regions more explicitly. However, the improvement of character localization is not clearly presented in word-level evaluation. Therefore, in order to show individual character localization ability of the detector, we take advantage of the pseudo-character box generation process in the CRAFT detector. When generating pseudo-ground-truths, supervision network calculates the difference between the number of generated pseudo characters with the number of ground-truth characters in the word transcription. Table. 6 shows number of *character error length* on each dataset measured with fully trained networks.

Dataset	Total Lengths	SynthText	w.o R-loss	w. R-loss	Diff.
ICDAR2015	46,107	9,324	1,251	1,147	-104 (-8.3%)
TotalText	53,645	16,385	5,050	4,521	- 529 (-10.4%)

Table 6. Comparison of *character error length* on each dataset with trained networks.

When training the network on SynthText dataset, *character error length* on each dataset is large. Error decreases further as training is performed on real datasets, but the value drops further by propagating recognition loss to the detection stage. This implies that the use of *Character Region Attention* improves the quality of the localization ability of the detector.

Importance of orientation estimation The orientation estimation is important because there are many oriented texts in scene text images. Our pixel-wise averaging scheme is very useful for the recognizer to receive well-defined features. We compare the results of our model when the orientation information is not used. On the IC15 dataset, the performance drops from 74.9% to 74.1% (-0.8%), and on TotalText dataset, the h-mean value drops from 78.7% to 77.5% (-1.2%). The results show that the use of accurate angle information escalates performance on rotated texts.

4.5 Discussions

Inference speed Since inference speed varies depending on the input image size, we measure the FPS on different input resolutions, each having a longer side of 960, 1280, 1600, and 2560. The test results give FPS of 9.9, 8.3, 6.8, and 5.4, respectively. For all experiments, we use Nvidia P40 GPU with Intel(R) Xeon(R) CPU. When compared with the 8.6 FPS of the VGG based CRAFT detector[2], the ResNet based CRAFTS network achieves higher FPS on the same sized input. Also, directly using the control points from the rectification module alleviates the need of post processing for polygon generation.

Fig. 9. Failure cases in IC15 dataset due to granularity difference. Red boxes denote detection results, cyan boxes denote ground truths.

Granularity difference issue We assume that the granularity difference between the ground-truth and prediction box causes relatively low detection performance on the IC15 dataset. Character-level segmentation methods tend to generalize character connectivity based on space and color cues, and not capture the whole feature of word instance. For this reason, the outputs do not follow the annotation style of the boxes required by the benchmark. The figure 9 shows the failure cases in the IC15 dataset, which proves that the detection results are marked incorrect while we observe acceptable qualitative results.

5 Conclusion

In this paper, we present an end-to-end trainable single pipeline model that tightly couples detection and recognition modules. *Character region attention* in the sharing stage fully exploit *character region map* to help recognizer rectify and attend better to the text regions. Also, we design the recognition loss propagate through detection stage and enhances the character localization ability of the detector. In addition, the rectification module in the sharing stage enables fine localization of curved texts, and obviates the need of developing hand crafted post-processing. The experimental results validate state-of-the-art performance of CRAFTS on various datasets.

References

1. Baek, J., Kim, G., Lee, J., Park, S., Han, D., Yun, S., Oh, S.J., Lee, H.: What is wrong with scene text recognition model comparisons? dataset and model analysis. In: The IEEE International Conference on Computer Vision (ICCV) (October 2019)
2. Baek, Y., Lee, B., Han, D., Yun, S., Lee, H.: Character region awareness for text detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 9365–9374 (2019)
3. Busta, M., Neumann, L., Matas, J.: Deep textspotter: An end-to-end trainable scene text localization and recognition framework. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2204–2212 (2017)
4. Busta, M., Patel, Y., Matas, J.: E2e-mlt-an unconstrained end-to-end method for multi-language scene text. In: Asian Conference on Computer Vision. pp. 127–143. Springer (2018)
5. Cheng, Z., Bai, F., Xu, Y., Zheng, G., Pu, S., Zhou, S.: Focusing attention: Towards accurate text recognition in natural images. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 5076–5084 (2017)
6. Cheng, Z., Xu, Y., Bai, F., Niu, Y., Pu, S., Zhou, S.: Aon: Towards arbitrarily-oriented text recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5571–5579 (2018)
7. Ch'ng, C.K., Chan, C.S.: Total-text: A comprehensive dataset for scene text detection and recognition. In: ICDAR. vol. 1, pp. 935–942. IEEE (2017)
8. Chng, C.K., Liu, Y., Sun, Y., Ng, C.C., Luo, C., Ni, Z., Fang, C., Zhang, S., Han, J., Ding, E., et al.: Icdar2019 robust reading challenge on arbitrary-shaped text-rrc-art. In: 2019 International Conference on Document Analysis and Recognition (ICDAR). pp. 1571–1576. IEEE (2019)
9. Deng, D., Liu, H., Li, X., Cai, D.: Pixellink: Detecting scene text via instance segmentation. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
10. Feng, W., He, W., Yin, F., Zhang, X.Y., Liu, C.L.: Textdragon: An end-to-end framework for arbitrary shaped text spotting. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 9076–9085 (2019)
11. Gao, Y., Chen, Y., Wang, J., Lei, Z., Zhang, X.Y., Lu, H.: Recurrent calibration network for irregular text recognition. arXiv preprint arXiv:1812.07145 (2018)
12. Gupta, A., Vedaldi, A., Zisserman, A.: Synthetic data for text localisation in natural images. In: CVPR. pp. 2315–2324 (2016)
13. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: ICCV. pp. 2980–2988. IEEE (2017)
14. He, T., Tian, Z., Huang, W., Shen, C., Qiao, Y., Sun, C.: An end-to-end textspotter with explicit alignment and attention. In: CVPR. pp. 5020–5029 (2018)
15. He, W., Zhang, X.Y., Yin, F., Liu, C.L.: Deep direct regression for multi-oriented scene text detection. In: CVPR. pp. 745–753 (2017)
16. Hu, H., Zhang, C., Luo, Y., Wang, Y., Han, J., Ding, E.: Wordsup: Exploiting word annotations for character based text detection. In: ICCV (2017)
17. Huang, Z., Zhong, Z., Sun, L., Huo, Q.: Mask r-cnn with pyramid attention network for scene text detection. In: 2019 IEEE Winter Conference on Applications of Computer Vision (WACV). pp. 764–772. IEEE (2019)
18. Jaderberg, M., Simonyan, K., Zisserman, A., et al.: Spatial transformer networks. In: Advances in neural information processing systems. pp. 2017–2025 (2015)

19. Karatzas, D., Gomez-Bigorda, L., Nicolaou, A., Ghosh, S., Bagdanov, A., Iwamura, M., Matas, J., Neumann, L., Chandrasekhar, V.R., Lu, S., et al.: Icdar 2015 competition on robust reading. In: ICDAR. pp. 1156–1160. IEEE (2015)
20. Karatzas, D., Shafait, F., Uchida, S., Iwamura, M., i Bigorda, L.G., Mestre, S.R., Mas, J., Mota, D.F., Almazan, J.A., De Las Heras, L.P.: Icdar 2013 robust reading competition. In: ICDAR. pp. 1484–1493. IEEE (2013)
21. Lee, C.Y., Osindero, S.: Recursive recurrent nets with attention modeling for ocr in the wild. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2231–2239 (2016)
22. Li, H., Wang, P., Shen, C.: Towards end-to-end text spotting in natural scenes. arXiv preprint arXiv:1906.06013 (2019)
23. Liao, M., Lyu, P., He, M., Yao, C., Wu, W., Bai, X.: Mask textspotter: An end-to-end trainable neural network for spotting text with arbitrary shapes. IEEE transactions on pattern analysis and machine intelligence (2019)
24. Liao, M., Shi, B., Bai, X.: Textboxes++: A single-shot oriented scene text detector. Image Processing **27**(8), 3676–3690 (2018)
25. Liao, M., Zhu, Z., Shi, B., Xia, G.s., Bai, X.: Rotation-sensitive regression for oriented scene text detection. In: CVPR. pp. 5909–5918 (2018)
26. Liu, J., Liu, X., Sheng, J., Liang, D., Li, X., Liu, Q.: Pyramid mask text detector. arXiv preprint arXiv:1903.11800 (2019)
27. Liu, W., Chen, C., Wong, K.Y.K.: Char-net: A character-aware neural network for distorted scene text recognition. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
28. Liu, W., Chen, C., Wong, K.Y.K., Su, Z., Han, J.: Star-net: a spatial attention residue network for scene text recognition. In: BMVC. vol. 2, p. 7 (2016)
29. Liu, X., Liang, D., Yan, S., Chen, D., Qiao, Y., Yan, J.: Fots: Fast oriented text spotting with a unified network. In: CVPR. pp. 5676–5685 (2018)
30. Long, S., He, X., Ya, C.: Scene text detection and recognition: The deep learning era. arXiv preprint arXiv:1811.04256 (2018)
31. Long, S., Ruan, J., Zhang, W., He, X., Wu, W., Yao, C.: Textsnake: A flexible representation for detecting text of arbitrary shapes. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 20–36 (2018)
32. Lyu, P., Liao, M., Yao, C., Wu, W., Bai, X.: Mask textspotter: An end-to-end trainable neural network for spotting text with arbitrary shapes. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 67–83 (2018)
33. Nayef, N., Patel, Y., Busta, M., Chowdhury, P.N., Karatzas, D., Khlif, W., Matas, J., Pal, U., Burie, J.C., Liu, C.l., et al.: Icdar2019 robust reading challenge on multi-lingual scene text detection and recognition—rrc-mlt-2019. In: 2019 International Conference on Document Analysis and Recognition (ICDAR). pp. 1582–1587. IEEE (2019)
34. Qin, S., Bissacco, A., Raptis, M., Fujii, Y., Xiao, Y.: Towards unconstrained end-to-end text spotting. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4704–4714 (2019)
35. Shi, B., Bai, X., Yao, C.: An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. IEEE transactions on pattern analysis and machine intelligence **39**(11), 2298–2304 (2016)
36. Shi, B., Wang, X., Lyu, P., Yao, C., Bai, X.: Robust scene text recognition with automatic rectification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4168–4176 (2016)

37. Shi, B., Yang, M., Wang, X., Lyu, P., Yao, C., Bai, X.: Aster: An attentional scene text recognizer with flexible rectification. IEEE transactions on pattern analysis and machine intelligence **41**(9), 2035–2048 (2018)
38. Shi, B., Yao, C., Liao, M., Yang, M., Xu, P., Cui, L., Belongie, S., Lu, S., Bai, X.: Icdar2017 competition on reading chinese text in the wild (rctw-17). In: 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR). vol. 1, pp. 1429–1434. IEEE (2017)
39. Shrivastava, A., Gupta, A., Girshick, R.: Training region-based object detectors with online hard example mining. In: CVPR. pp. 761–769 (2016)
40. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR (2015)
41. Sun, Y., Ni, Z., Chng, C.K., Liu, Y., Luo, C., Ng, C.C., Han, J., Ding, E., Liu, J., Karatzas, D., et al.: Icdar 2019 competition on large-scale street view text with partial labeling-rrc-lsvt. In: 2019 International Conference on Document Analysis and Recognition (ICDAR). pp. 1557–1562. IEEE (2019)
42. Sun, Y., Zhang, C., Huang, Z., Liu, J., Han, J., Ding, E.: Textnet: Irregular text reading from images with an end-to-end trainable network. In: Asian Conference on Computer Vision. pp. 83–99. Springer (2018)
43. Wang, W., Xie, E., Li, X., Hou, W., Lu, T., Yu, G., Shao, S.: Shape robust text detection with progressive scale expansion network. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 9336–9345 (2019)
44. Xing, L., Tian, Z., Huang, W., Scott, M.R.: Convolutional character networks. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 9126–9136 (2019)
45. Xu, Y., Wang, Y., Zhou, W., Wang, Y., Yang, Z., Bai, X.: Textfield: Learning a deep direction field for irregular scene text detection. IEEE Transactions on Image Processing (2019)
46. Zhan, F., Lu, S.: Esir: End-to-end scene text recognition via iterative image rectification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2059–2068 (2019)
47. Zhang, C., Liang, B., Huang, Z., En, M., Han, J., Ding, E., Ding, X.: Look more than once: An accurate detector for text of arbitrary shapes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 10552–10561 (2019)
48. Zhong, Z., Sun, L., Huo, Q.: An anchor-free region proposal network for faster r-cnn-based text detection approaches. International Journal on Document Analysis and Recognition (IJDAR) **22**(3), 315–327 (2019)
49. Zhou, X., Yao, C., Wen, H., Wang, Y., Zhou, S., He, W., Liang, J.: East: an efficient and accurate scene text detector. In: CVPR. pp. 2642–2651 (2017)