applied sciences

MDPI

Article

CommuSpotter: Scene Text Spotting with Multi-Task Communication

Liang Zhao *, Greg Wilsbacher and Song Wang

Computer Science & Engineering Department, University of South Carolina, Columbia, SC 29201, USA; gregw@mailbox.sc.edu (G.W.); songwang@cec.sc.edu (S.W.)
* Correspondence: lz4@email.sc.edu

Abstract: Scene text spotting is a challenging multi-task modulation for locating and recognizing texts in complex scenes. Existing end-to-end text spotters generally adopt sequentially decoupled multi-tasks, consisting of text detection and text recognition modules. Although customized modules are designed to connect the tasks closely, there is no interaction among multiple tasks, resulting in compatible information loss for the overall text spotting. Moreover, the independent and sequential modulation is unidirectional, accumulating errors from early to later tasks. In this paper, we propose CommuSpotter, which enhances multi-task communication by explicitly and concurrently sharing compatible information in overall scene text spotting. To address task-specific inconsistencies, we propose a Conversation Mechanism (CM) to extract and exchange expertise in each specific task with others. Specifically, the detection task is rectified by the text recognition task to filter out duplicated results and false positives, while the text recognition task is corrected by the rectified text detection task to replenish missing characters and decrease non-text interruptions. Consequently, the communication compensates for interaction information and breaks the sequential pipeline of error propagation. In addition, we adopt text semantic segmentation in the text recognition task, which reduces the complex design of customized modules and corresponding extra annotations. Compared with state-of-the-art methods, experimental results show that our method achieves competitive results with computation efficiency.

Keywords: scene text spotter; end-to-end text spotting; text detection; text recognition

Citation: Zhao, L.; Wilsbacher, G.; Wang, S. CommuSpotter: Scene Text Spotting with Multi-Task Communication. *Appl. Sci.* **2023**, *13*, 12540. https://doi.org/10.3390/app132312540

Academic Editor: Andrea Prati

Received: 23 August 2023
Revised: 13 November 2023
Accepted: 19 November 2023
Published: 21 November 2023

1. Introduction

Scene text spotting aims to detect and recognize various texts in complex scenes simultaneously. Derived from general object detection, scene text detection faces many challenges, such as scattered and changing characters in different word detection, text style and font variations, background interruptions, etc. On the other hand, scene text recognition also incorporates language processing techniques to decode sequence features for text prediction. The end-to-end training and application of scene text detection and recognition have garnered increased research interest in recent years. It not only integrates the two main tasks of text detection and recognition, but it also bridges the training schemas of different tasks for optimization. It has wide applications in artificial intelligence and computer vision, such as criminal investigation [1], video information retrieval [2], robotic assistance [3], and autonomous driving [4]. Modern scene text spotting comes up with challenges, such as artificially embellished texts, various text perspectives, and complex text shapes.

Existing two-stage paradigm text spotters conduct text detection tasks for text localization and text recognition tasks for words, while one-stage paradigm text spotters remove text detection but introduce customized modules for connecting the backbone with text recognition modules. The configuration examples of typical text spotters are collected in Table 1. Transformer-based backbones and text detectors greatly simplify proposal

generation and boost performance at the expense of the computation cost. The customized modules are specifically designed to shrink the text instances or refine the features.

Although current end-to-end scene text spotters [5–12] have achieved substantial progress, there are three limitations. Firstly, the information or expertise from various multi-tasks compensates for each other in the final text spotting, but some expertise is lost in the current independent multi-task methods. For example, many studies improve information usage by sharing the network backbone [9,12,13], integrating the recognition loss [14,15], or bridging customized modules [5–8], as shown in Figure 1a. However, the interaction between tasks is not adequately identified. As indicated by the yellow arrow, there is only compensation from the text detection task (or customized modules) to the text recognition task, but not inversely, where the expertise from customized modules or text recognition is scarcely utilized to interact with early text detection tasks or proposal generations. There is no communication or usage of each other's information for multi-tasks in the text-spotting process. Recently, some text spotters [14,15] have adopted a Transformer to understand the relationship between text instances but not between tasks. As a result, independent multi-task modulation leads to the loss of concurrent compensatory expertise from multiple tasks in text spotting. Secondly, apart from task-specific information loss, pipeline errors accumulate in current unidirectional text spotters. Errors from early tasks are not identified and, thus, accumulate in later tasks, leading to back-and-forth training. During inference, the methods become unreliable from the start when there is no ground truth in each step. Thirdly, the sophisticated customized modules increase model complexity and require expensive extra annotations. For example, some customized modules involve designs and labels for text lines [16], text strokes [17], text center points [18], and so on.

Table 1. Comparison between typical models. There are mainly two categories of methods divided by Transformer architecture. "CNN" is the traditional ResNet backbone. "Trans" stands for Transformer architecture. "Seg" is equal to "segmentation" for short. "Att" is the attention-based loss, while "CTC" is the CTC loss. Specific module techniques and details can be found in each method.

Category	Methods	Backbone	Proposals	Customized	Seg	Rec
CNN	Qin et al. [5]	CNN	RPN	RoI Masking	Instance	Att
	Mask TextSpotter v2 [19]	CNN	RPN	Box Detection	Instance	Att
	Mask TextSpotter v3 [6]	CNN	-	Seg Proposals	Instance	Att
	ABCNet [8]	CNN	-	Bezier Curve	Instance	CTC
	MANGO [7]	CNN	-	Mask Attention	Instance	Att
Trans	SwinTextSpotter [14]	CNN + Trans	Query Box	Recognition Conversion	Instance	Att
	TextTranSpotter [15]	CNN + Trans	Query Box	Hungarian Match	Instance	Att

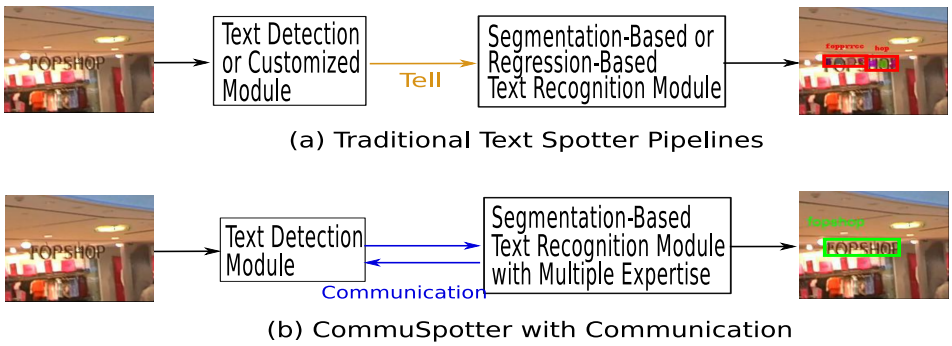

Figure 1. An illustration of typical text spotters (**a**); frameworks compared with our proposed spotter with communication mechanisms in multi-tasks (**b**).

In this work, we propose CommuSpotter, which explicitly and concurrently communicates multi-task expertise with others. Multi-task communication involves information

from all tasks complementing and assisting each individual task. Drawing inspiration from the image retrieval task [11], we develop a Conversation Mechanism (CM) to embed expertise vectors for multi-task interaction. Thus, this communication of expertise facilitates information exchanges among text detection and recognition modules, as shown in Figure 1b, breaking the unidirectional pipeline and reducing accumulated errors from the beginning. Additionally, we introduce text semantic segmentation for text recognition without requiring further annotations. This is achieved by adopting the independently pre-trained module weights as priors, replacing complex customized modules and their extra annotations. Finally, CommuSpotter constructs a concise framework and achieves fast convergence. The contributions of this paper are threefold: (1) Instead of independently and sequentially performing text detection (or customized modules) and text recognition multi-tasks, we developed CommuSpotter, which facilitates explicit communication through the designed Conversation Mechanism (CM). This mechanism embeds task-specific expertise concurrently across all tasks, compensating for task-specific information or expertise and reducing error propagation throughout the entire text-spotting process. (2) We employ text semantic segmentation expertise for text recognition tasks, reducing the need for complex custom module designs and their associated costly annotations. (3) We conduct comprehensive experiments on multiple text datasets. The comparisons between existing approaches demonstrate the advantages of our proposed method.

2. Related Work

2.1. Scene Text Spotter of Two-Stage Paradigm

To address the challenges of arbitrary texts, Lyu et al. [13] developed Mask TextSpotter v1, which includes long short-term memory (LSTM) [20] in text recognition tasks to boost spotting results. In Mask TextSpotter v2 [19], text and character instance segmentations are adopted to improve recognition performance. Qiao et al. [21] predicted additional latent information from text detection tasks to enhance text instance segmentation. Qin et al. [5] developed Regions of Interest (RoIs) masking to improve segmentation accuracy by selecting and fusing features for instances. These studies focused on improving the text recognition task. Some of the following focus on improving the early text detection task. FOTS [9] and TextNet [22] adopt rotating RoIs and perspective RoIs to handle irregular and multi-oriented text detections, respectively. CRAFT [23] groups character region features from the text detector to reinforce character attention for the recognizer. Kittenplon et al. [15] adopted Transformer to improve the representation from the shared backbone, while text detection and recognition tasks were parallel and independent. SwinTextSpotter [14] adopts the Swin Transformer [24] in the backbone for better text representations. The expertise from the text detection task is transferred to the text recognition task, but there is no inverse interaction. In addition, errors accumulate from early tasks to later tasks. Furthermore, some dataset annotations are tailored to specific methods, such as character and polygon annotations [13,25], text-line annotations [16], etc.

2.2. Scene Text Spotter of One-Stage Paradigm

Many customized modules have been developed to replace traditional text detection with bounding boxes, thereby building a closer connection between multiple tasks. For some segmentation-based methods, Mask TextSpotter v3 [6] uses segmentation proposals for arbitrarily shaped text recognition. Liu et al. [8,26] designed Bezier curves to represent text instances. MANGO [7] extracts text center lines and text and character segmentation maps for text grouping and recognition. For some regression-based methods, Wang et al. [27] designed the instance boundary points to improve text instance shapes. TextDragon [28] combines sliding RoIs with local points. SRSTS [29] adopts anchor points in text recognition. These methods replace bounding boxes with accurate text boundaries for text recognition, but customized modules are also little rectified by later expertise in text recognition, similar to two-stage paradigms. Again, the errors are propagated. Customized modules also require extra annotations, such as character annotations [6], stroke annotations [17], and so on.

2.3. Scene Text Spotter with Back-Propagation

Some studies show backward compensation among tasks but only build conversion by recognition loss. Zhong et al. [30] adopted a spatial transform network (STN) to propagate the recognition loss back to the text detection task. SwinTextSpotter [14] proposes the synergy mechanism for joint optimization by backpropagating the recognition loss. However, the loss function does not facilitate concurrent expertise interaction among sequential tasks, and it is not effective during testing in encoding representations.

3. Text Spotter with Communication

Scene text spotting mainly consists of text detection and recognition tasks, as shown in Figure 1. For the text detection task, the image is fed into a CNN-based network to extract feature maps, which are then pooled by $MaxPooling$ and $Softmax$ layers for the coordinate prediction of text locations and classification of text detection. The metrics include accuracy, recall, and F-score of text classification, as well as location overlap with the ground truths. Due to challenges in scene text variations, detection often contains many false positives, leading to incorrect recognition based on those detections. Text recognition relies on RNN networks that sequentially process the feature maps to predict each word character in text instances. For a maximum length of 25 for each word, each character is predicted and grouped to form the final recognition results. The metrics include recognition accuracy, recall, and F-score for every word.

3.1. Architecture

The whole framework of CommuSpotter is shown in Figure 2. It contains text detection and text recognition modules. Given an image, I, a backbone of ResNet-50 is used to extract feature maps, P, denoted as $\{P_2, P_3, P_4, P_5, P_6\}$, with a Feature Pyramid Network (FPN) following [31]. The Region Proposal Network (RPN) generates some Region of Interest (RoI) proposals, R, in five scales of $\{32^2, 64^2, 128^2, 256^2, 512^2\}$ on the pyramid features. The redundant candidates are filtered out by Non-Maximum Suppression (NMS). The pyramid features, P, and proposals, R, are processed by the RoIAlign [32] layer to generate the RoI features, F_{RoI} and F'_{RoI}, which are then fed into the text detection module and text recognition modules. Moreover, the semantic expertise, S and S', from the feature maps are extracted via softmax operations and attention-based refinements.

In the text detection module, the RoI features, F_{RoI}, interact with the expertise from the text recognition module through the specially designed Conversation Mechanism (CM) (in Section 3.2) to generate meaningful text detection through the classification layer. In the text recognition module, we adopt a segmentation-based method for text instance segmentation as the word expertise, as described in Section 2.2). Additionally, we introduce text semantic segmentation expertise S and S' as character details for text recognition. The representation is communicated by our designed Conversation Mechanism (CM) (in Section 3.3). Finally, text sequences are recognized by the RNN mechanism [33]. Note that the contents in this Section 3.1 are existing pipelines of scene text spotting, Sections 3.2 and 3.3 below are our proposed approaches.

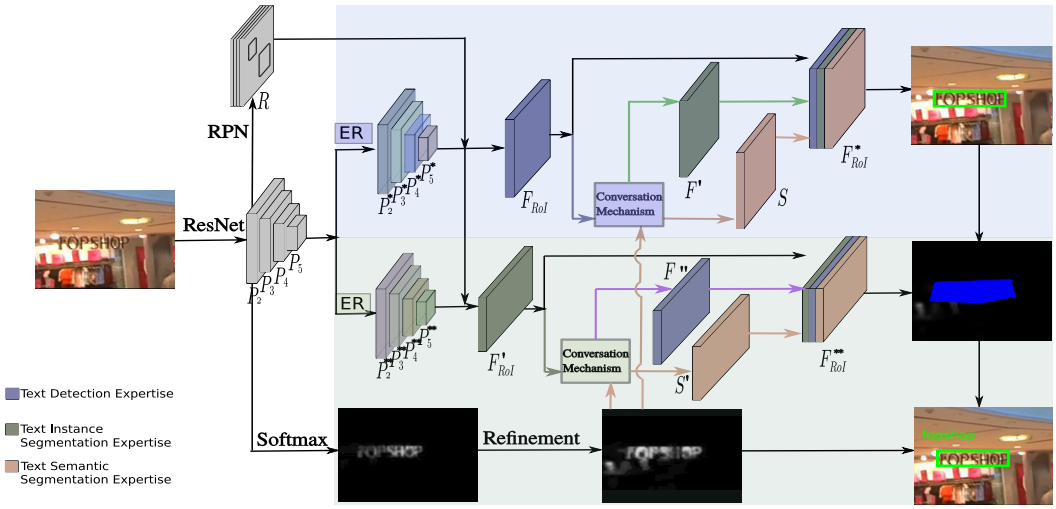

Figure 2. Architecture of the proposed CommuSpotter. From the backbone, the pyramid features, P, and the proposals, R, are processed by text detection and text recognition modules. "ER" denotes Extensive Representation, and Conversation Mechanisms are designed for different communications. From the backbone, text semantic segmentation expertise is generated for the text recognition module. Multiple forms of expertise interact with each other concurrently. The black arrows indicate the forward flow of information, while the colorful arrows represent the backward interaction of multi-task expertise.

3.2. Text Detection Communication

Generally, the modality interaction [34] is either conducted by concatenating features and interacting together or by directly conducting interactions among features. Different tasks generate information from different perspectives for the final text spotting in text spotters. Existing text detection and recognition modules share the same backbone for different tasks. Considering the modeling efficiency, we keep the same backbone but introduce Extensive Representation (ER) on pyramid features for different tasks, as shown in Figure 3a. Prior to applying RoIAlign, as detailed in [32], which builds local coherence for feature alignments, we embed high-level pyramid features on a large receptive field for text detection. Specifically, the shared pyramid features P_2 to P_5 are updated as follows:

$$P^*_{i-1} = Gcm(Ur(P_i) \cdot P_{i-1}), i = 3, 4, 5, \tag{1}$$

where P_2 is the shape of $C \times H/4 \times W/4$, P_3 is the shape of $C \times H/8 \times W/8$, P_4 is the shape of $C \times H/16 \times W/16$, and P_5 is the shape of $C \times H/32 \times W/32$. C represents the dimension of channels, set at 256, while H and W denote the input resolutions in terms of height and width, respectively. The operation $Ur(\cdot)$ is used for up-sampling. $Gcm(\cdot)$ denotes a sequence of convolution layers, consisting of 1×3 and 3×1 convolutions. They are applied to reduce dimensionality gradually. Finally, the results are combined as P^*, consisting of P^*_2 to P^*_5.

Compared with the general object detection of integrated targets, split characters inside texts always cause false positive detection results due to background interruptions and character-like non-texts. To achieve concurrent semantic guidance, we developed a Conversation Mechanism (CM) through a series of straightforward operations. This mechanism incorporates the expertise of a text recognition module, comprising (1) text semantic segmentation expertise S (as detailed in Section 3.3) to identify character objects (other than non-texts) and (2) text instance segmentation expertise F' to construct exact locations. Specifically, the refined features, P^*, and proposals, R, are fed into the RoIAlign with text semantic segmentation expertise S to generate the aligned RoI feature F_{RoI} of the shape $N \times C \times H_{RoI} \times W_{RoI}$, where N is the number of mapped proposals, and H_{RoI} and W_{RoI} are feature map resolutions set to 7. To obtain text instance segmentation expertise F', we

conducted a sequence of four convolutional layers *Conv* and one transposed convolutional layer *TrConv* on the RoI features F_{RoI}. Then, the expertise was combined.

$$F_{RoI} = RoIAlign(P^*, R) + S, \tag{2}$$

$$F' = TrConv(Conv(F_{RoI})), \tag{3}$$

$$F^*_{RoI} = F' + F_{RoI}. \tag{4}$$

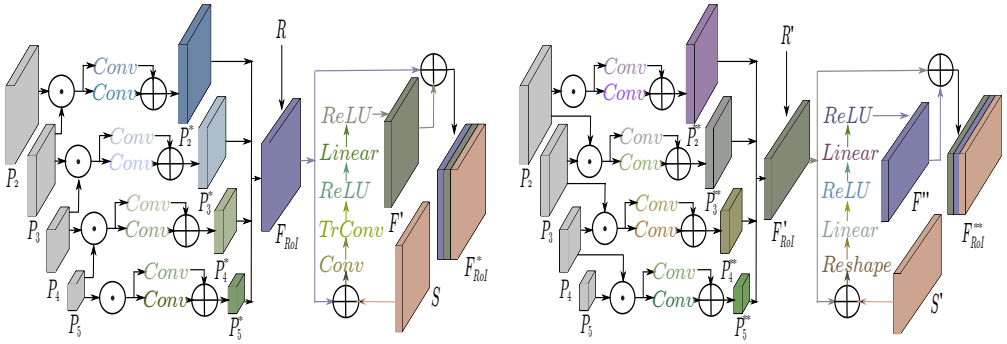

Figure 3. The communication networks of (**a**) the text detection module and (**b**) text recognition module.

3.3. Text Recognition Communication

As mentioned above, a segmentation-based text recognition module adopts text or character instance segmentation. Similar to the text detection module, the shared features are first extended by the reverse Extensive Representation (ER) to embed global information, as shown in Figure 3b. Specifically, the filtered pyramid features, P, consisting of multiple levels, from P_2 to P_5, are presented for the module as follows:

$$P^{**}_{i+1} = Gcm(Dr(P_i) \cdot P_{i+1}), i = 2, 3, 4, \tag{5}$$

where $Dr(\cdot)$ is the downsampling operation.

The instance segmentation-based recognition methods always suffer from the problem of missing characters in texts, especially when some characters inside a text are artistically designed with different textures, shapes, fonts, etc. From this view, the purpose of the Conversation Mechanism (CM) is to integrate (1) the renewed expertise from text detection module F'' as global guidance and (2) text semantic segmentation expertise S', which provides local details. Specifically, the extensive features, P^{**}, and filtered proposals, R', from RPN are fed into the RoIAlign with text semantic segmentation expertise S', to generate the aligned RoI feature, F'_{RoI}, of the shape $N' \times C \times H'_{RoI} \times W'_{RoI}$, where N' is the number of instances, and H'_{RoI} and W'_{RoI} are the resolutions of 16 and 64, respectively.

$$F'_{RoI} = RoIAlign(P^{**}, R') + S', \tag{6}$$

$$F'' = Lr(Lr(F'_{RoI})), \tag{7}$$

$$F^{**}_{RoI} = F'' + F'_{RoI}. \tag{8}$$

Then, the aligned RoI feature, F'_{RoI}, is transferred by two consequent linear convolutional layers $Lr(\cdot)$. Finally, we concatenate text detection expertise F'' with the RoI features F'_{RoI}.

Another problem of segmentation-based recognition is that character details are ignored in text spotters. The pixel-level expertise of characters can provide information to differentiate characters of similar shapes. For example, "hot" and "hat" only differ in one character's details. Due to the lack of pixel-level annotations for scene text datasets, text semantic segmentation is not applied in existing scene text spotters. Instead of designing complex modules to tighten the character regions with extra annotations, we introduce text semantic segmentation [35] expertise S and expertise S' with individually pre-trained

weights. The initial semantic segmentation results, S_e, are refined by the attention mechanism [36] to obtain a better text semantic segmentation S_{re}, as shown in Figure 2.

$$S_f = ResNet(I), \tag{9}$$

$$S_e = Softmax(S_f), \tag{10}$$

$$S_{att} = Att(S_e, S_f), \tag{11}$$

$$S_{re} = Convolution(S_e, S_{att}), \tag{12}$$

where S_f is the fusion of pyramid features P_2 to P_5, $Softmax$ is the softmax operation, Att is an attention layer of the dot-product operation, and $Convolution$ is a series of convolution layers of kernel size 5×5 and 1×1, to fuse the attention with the feature maps. To communicate this expertise in the text spotter, we take pooling operations on text semantic segmentation S_{re} with different resolutions. Then, text semantic segmentation expertise S and expertise S' are generated.

$$S = \mathcal{P}(S_{re}), \tag{13}$$

$$S' = \mathcal{P}'(S_{re}), \tag{14}$$

where $\mathcal{P}$ and $\mathcal{P}'$ are the pooling layers with different resolutions. The shape of expertise S is $N \times C \times H_{RoI} \times W_{RoI}$, and S' is $N' \times C \times H'_{RoI} \times W'_{RoI}$.

3.4. Optimization

The whole framework loss $\mathcal{L}$ is defined as follows:

$$\mathcal{L} = \mathcal{L}_{rpn} + \alpha \mathcal{L}_{rcnn} + \beta \mathcal{L}_{rec}, \tag{15}$$

$$\mathcal{L}_{rcnn} = \mathcal{L}_{cls} + \mathcal{L}_{reg}, \tag{16}$$

$$\mathcal{L}_{rec} = \mathcal{L}_{ins} + \delta \mathcal{L}_{seg} + \epsilon \mathcal{L}_{seq}, \tag{17}$$

where $\mathcal{L}_{rpn}$, $\mathcal{L}_{rcnn}$, and $\mathcal{L}_{rec}$ are the losses of RPN [13], text detection module, and text recognition module, respectively. The weights of α and β are equal to 1.0 and 1.0, respectively. The detection module loss is $\mathcal{L}_{rcnn}$, consisting of the cross-entropy classification loss $\mathcal{L}_{cls}$ and the smooth L1 regression loss $\mathcal{L}_{reg}$ [37]. The text recognition module loss $\mathcal{L}_{rec}$ includes a cross-entropy text instance segmentation loss $\mathcal{L}_{ins}$, a character instance segmentation loss $\mathcal{L}_{seg}$, and a sequence recognition loss $\mathcal{L}_{seq}$ [13]. The weights of δ and ϵ are empirically set to 1.0 and 0.2, respectively. The $\mathcal{L}_{seq}$ follows a summation of the logarithm loss [13]. The text instance map is encoded by convolutional and max pooling layers with two-dimensional representations [38], and fed into a seq2seq recognizer [39] to generate text sequences.

4. Experiments

4.1. Datasets

SynthText is a synthetic dataset [40] of around 800 k images with comprehensive text samples used for pre-training. ICDAR2013 (IC13) is for the 2013 Robust Reading Competition [41], consisting of 229 training images and 233 test images. ICDAR2015 (IC15) is provided by the 2015 Robust Reading Competition [42]. It contains incidental scene texts of 1000 training images and 500 test samples. Total-Text (TT) [43] focuses on arbitrarily shaped texts, including 1255 training and 300 test images. SCUT [44] contains 1162 natural images from Flickr [13]. These real data are used for fine-tuning the model. The evaluation is conducted on IC15 and TT, current scene text-spotting benchmarks.

4.2. Implementation Details

The model is trained using PyTorch with two Tesla-V100 GPUs and tested on a single GPU. Following the Mask TextSpotter v2 [19], the training process consists of two parts: pre-training and fine-tuning. The optimizer is Stochastic Gradient Descent (SGD), set with a weight decay equal to 0.001 and a momentum of 0.9. In the pre-training stage, the model

is trained on the SynthText [40] for 270 K iterations. The initial learning rate is set to 0.01 and decayed at every 90 K iterations by a tenth. In the fine-tuning stage, the model is trained on multiple real-world image datasets for 90 K iterations. The learning rate is set to 0.001. In the inference stage, the input images are fed into the model to generate proposals, instances, and recognition predictions.

4.3. Ablation Study

The experiments adopt percentage values of the end-to-end recognition F-scores on the ICDAR15 dataset with a strong lexicon. We train a model that only adopts the Extensive Representation (ER) mechanism from the baseline Mask TextSpotter v2 [19]. The ER improves the recognition performance from 82.1% to 82.5% in Table 2.

Table 2. End-to-end recognition results of ICDAR15 on different model configurations. "CM-P" stands for part of the Conversation Mechanism without text semantic segmentation expertise.

Methods	Extensive Representation	CM-P	Conversation Mechanism	F-Score
E2E-baseline				82.1
w/ER	✓			82.5
w/CM-P		✓		83.7
w/CM-P	✓	✓		84.6
Full Setting	✓	✓	✓	85.8

To validate the effectiveness of communication between text detection and instance segmentation tasks, we add the Conversation Mechanism (CM) to them but without expertise from text semantic segmentation. The recognition results indicate that concurrent communication can improve performance from 82.1% to 83.7%. If equipped with the above ER, it can achieve 84.6%. Finally, with full settings of CM, we add the interaction from the text semantic segmentation. The F-score is further improved from 84.6% to 85.8%. Thus, there is a collaborative effect for the entire text spotter.

We train and fine-tune several models of the configurations on the Total-Text dataset, as shown in Table 3. The ER improves the spotting performance from the baseline Mask TextSpotter v2 [19] of 77.4% to 78.4%. The singular CM-P can improve the performance from 77.4% to 80.3%. Together with ER, the recognition results indicate that concurrent communication can improve performance to 81.7%. With full settings of CM, we add the interaction from text semantic segmentation. The F-score is further improved to 83.4%.

Table 3. End-to-end recognition results of Total-Text on different model configurations. "CM-P" stands for part of the Conversation Mechanism without text semantic segmentation expertise.

Methods	Extensive Representation	CM-P	Conversation Mechanism	F-Score
E2E-baseline				77.4
w/ER	✓			78.4
w/CM-P		✓		80.3
w/CM-P	✓	✓		81.7
Full Setting	✓	✓	✓	83.4

As mentioned above, some methods adopting Transformer or customized modules may train the model on extra datasets or make corresponding extra labels for the datasets. It causes difficulty in the fair comparison of different approaches. We compare the model efficiency in Table 4. For the abbreviation of datasets, "CST" is Curved SynthText [8]; "COCO" is COCO-Text [45]; "MLT" is ICDAR-MLT [46]; "IC13" is ICDAR2013 [41]; and "CTW" is SCUT-CTW1500 [47]. We compare the costs of computation resources with some approaches in Table 4. For the maximum iterations of convergence in the training process, we only need 360 K iterations in total, which is almost the fastest model. The GPU hour is roughly estimated from previous papers or re-implementation. Our GPU hour is a little more than the most

lightweight ABCNet [8], but our performance is better than that work. As a result, our method can achieve competitive performance at a lower cost compared to previous approaches. It is a good trade-off between model performance and computational efficiency.

Table 4. Comparison between training configuration and computation cost. In the pre-training, mix-training, and/or fine-tuning stages, the word "Data" represents the datasets used for training, and "Iter." denotes the number of convergence iterations needed. "GPU" denotes the estimated GPU hours for each method.

Methods	Pre-Train		Mix-Train or/and Fine-Tune		GPU
	Data	Iter.	Data	Iter.	
MANGO [7]	SynthText	600 K	CST, COCO, MLT, IC13, 1C15, Total	250 K	∼1344
ABCNet [8]	CST, COCO, MLT	260 K	Total, CTW	150 K	∼288
SwinTextSpotter [14]	CST, MLT, IC13, IC15, TT	450 K	IC13, IC15, Total, MLT, CTW	90 K	∼336
TTS [15]	SynthText	-	SynthText, IC13, IC15, Total, SCUT, COCO	-	∼1200
Mask TextSpotter v2 [19]	SynthText	270 K	SynthText, IC13, IC15, Total, SCUT	150 K	∼432
Ours	SynthText	270 K	IC13, IC15, Total, SCUT	90 K	∼312

4.4. Incidental Texts

All the following comparison metrics on the text-spotting datasets are percentage values. We evaluate the method on incidental texts of IC15 [42]. The evaluation results are shown in Table 5. Our method achieves a state-of-the-art recall of 88.3% and an F-score of 89.8% compared with previous studies. For the end-to-end recognition evaluation, our method outperforms the previous non-Transformer methods in the strong and generic lexicon. The F-score achieves 85.8% and 74.9%. Compared with recent Transformer based methods, it is hard to distinguish the effects of the Transformer from the modulation. Also, TextTranSpotter (TTS) [15] achieves a higher F-score for the generic lexicon when trained with 43 K more images compared with our 4 K images. Our results demonstrate the effectiveness of communication among multi-tasks for end-to-end text spotting in incidental texts. Apart from the Transformer, this represents a promising exploration into the simple and clean modulation of text spotters.

Table 5. Comparison results on the ICDAR2015 dataset. For the detection result, "P", "R", and "F" represent the metrics of precision, recall, and F-score, respectively. The end-to-end recognition evaluation is the F-score. "S", "W", and "G" means strong, weak, and generic lexicons, respectively.

Methods	Detection			End-to-End		
	P	R	F	S	W	G
FOTS [9]	91.0	85.2	88.0	81.1	75.9	60.8
Qin et al. [5]	89.4	85.8	87.5	83.4	79.9	67.9
Mask TextSpotter v1 [13]	91.6	81.0	86.0	79.3	73.0	62.4
Text Perceptron [21]	91.6	81.8	86.4	80.5	76.6	65.1
TextDragon [28]	92.5	83.8	87.9	82.5	78.3	65.2
CharNet [25]	91.2	88.3	89.7	80.1	74.5	62.2
ABCNet v2 [26]	-	-	-	82.7	78.5	73.0
CRAFTS [23]	89.0	85.3	87.1	83.1	**82.1**	74.9
Boundary [27]	89.8	87.5	88.6	79.7	75.2	64.1
Mask TextSpotter v2 [19]	86.6	87.3	87.0	83.0	77.7	73.5
Mask TextSpotter v3 [6]	-	-	-	83.3	78.1	74.2
MANGO [7]	-	-	-	81.8	78.9	67.3
SwinTextSpotter [14]	-	-	-	83.9	77.3	70.5
TTS [15]	-	-	-	85.2	81.7	**77.4**
GLASS [48]	-	-	-	84.7	80.1	76.3
SRSTS [29]	**96.1**	82.0	88.4	85.6	81.7	74.5
Ours	91.4	**88.3**	**89.8**	**85.8**	80.2	74.9

4.5. Arbitrary Texts

To verify the effectiveness of the model on irregular texts, we conduct experiments on the Total-Text (TT) [40], following the detection and end-to-end evaluation protocols publicized with the dataset. The comparison results are presented in Table 6. Our detection results achieve state-of-the-art performance compared with non-Transformer methods and achieve the best recall and F-scores compared to recent Transformer methods. The recognition results outperform previous non-Transformer text spotters while not as good as Transformer methods. However, the non-lexicon F-score of 73.0% is significantly improved from the baseline Mask TextSpotter v2 [19] of 65.3%. The recognition result of 83.4% with lexicon outperforms the corresponding baseline of 77.4%. Compared with SwinTextSpotter [14] of the ResNet backbone (shown as SwinTextSpotter-R), our recognition results achieve better performance. It shows that without a Transformer, our communication mechanism can be better in modulation. The newest SRSTS [29] achieves the best recognition results with Transformer decoders, which indicates that we still have improvement space in the recognition module.

Table 6. Comparison results on the Total-Text (TT) dataset. The detection results are measured by precision (P), recall (R), and F-score (F). The recognition F-scores include evaluation without lexicon as "None" and with lexicon as "Full".

Methods	Detection			End-to-End	
	P	R	F	None	Full
TextSnake [49]	82.7	74.5	78.4	-	-
Mask TextSpotter v1 [13]	69.0	55.0	61.3	52.9	71.8
TextNet [22]	68.2	59.5	63.5	54.0	-
Mask TextSpotter v2 [19]	81.8	75.4	78.5	65.3	77.4
Mask TextSpotter v3 [6]	-	-	-	71.2	78.4
Boundary [27]	85.2	83.5	84.3	65.0	76.1
CRAFTS [23]	89.5	85.4	87.4	78.7	-
ABCNet [8]	-	-	-	64.2	75.7
ABCNet v2 [26]	-	-	87.0	70.4	78.1
PAN++ [50]	-	-	-	68.6	78.6
MANGO [7]	-	-	-	71.7	82.6
SwinTextSpotter-R [14]	-	-	87.2	72.4	83.0
SwinTextSpotter [14]	-	-	88.0	74.3	84.1
GLASS [48]	-	-	-	76.6	83.0
SRSTS [29]	**92.0**	83.0	87.2	**78.8**	**86.3**
TTS [15]	-	-	-	75.6	84.4
Ours	90.4	**91.5**	**90.1**	73.0	83.4

4.6. Inference Speed

The inference speed is compared with previous approaches, as shown in Table 7. Not all recent studies provide inference time statistics. Compared to MANGO [7], our method is a little slower but with better performance. As mentioned above, our method is more efficient at training and can be easily used in practical applications without complex modules and expensive data labeling.

Table 7. Frames per second (FPS) comparison on different inference datasets.

Methods	ICDAR2015	Total-Text
Mask TextSpotter v2 [19]	3.1	-
Mask TextSpotter v3 [6]	2.5	-
ABCNet [8]	-	6.9
MANGO [7]	4.3	4.3
Ours	2.9	2.4

4.7. Qualitative Results

We can see the improvements in our approach (second row) compared with the baseline Mask TextSpotter v2 [19] (first row) from Figure 4. For the first two examples with complex scenes, the original method always catches false positive detections and obtains wrong recognition results. That is, decorated curves are wrongly detected as texts and recognized as non-existing information. For the last two examples of arbitrarily shaped texts, the text shapes and background interruptions always cause incomplete text instances. For example, the curved character layout is easily spotted as duplicated text but not whole instances. As a result, there are many unclear recognized meanings in the scene. There are fewer errors in our method in the second row. We present more qualitative samples in the third and fourth rows of Figure 4 for comparison.

Figure 4. Comparison of qualitative samples on ICDAR2015 and Total-Text datasets. The bounding boxes or polygons and recognized texts in red and pink colors are the wrong results, while those in green are the correct ones.

5. Conclusions

We proposed a streamlined framework to facilitate the exchange of expertise among multiple tasks in the scene text spotter. With the bidirectional communication between text detection and text recognition modules, our CommuSpotter allows for concurrent expertise exchange and early error correction. Instead of a complex, customized module design for tight character regions, we introduce text semantic segmentation in the recognition module. We conduct experiments on incidental and curved text datasets; the proposed method achieves consistently competitive performance with model efficiency.

Author Contributions: Conceptualization, L.Z., G.W. and S.W.; methodology, L.Z.; software, L.Z.; validation, L.Z; writing—original draft preparation, L.Z.; writing—review and editing, L.Z., G.W. and S.W.; supervision, S.W.; funding acquisition, G.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the XSEDE Program of the National Science Foundation and the Aspire-II Research Program at the University of South Carolina.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available in article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Al-Zaidy, R.; Fung, B.; Youssef, A.; Fortin, F. Mining criminal networks from unstructured text documents. *Digit. Investig.* **2012**, *8*, 147–160. [CrossRef]
2. Sivic, Z. Video Google: A text retrieval approach to object matching in videos. In Proceedings of the 9th IEEE International Conference on Computer Vision, Nice, France, 14–17 October 2003; pp. 1470–1477.
3. Looije, R.; Neerincx, M.; Cnossen, F. Persuasive robotic assistant for health self-management of older adults: Design and evaluation of social behaviors. *Int. J. Hum. Comput. Stud.* **2010**, *68*, 386–397. [CrossRef]
4. Jung, S.; Lee, U.; Jung, J.; Shim, D. Real-time Traffic Sign Recognition system with deep convolutional neural network. In 13th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI), Xi'an, China, 19–22 August 2016; pp. 31–34.
5. Qin, S.; Bissacco, A.; Raptis, M.; Fujii, Y.; Xiao, Y. Towards unconstrained end-to-end text spotting. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 4704–4714.
6. Liao, M.; Pang, G.; Huang, J.; Hassner, T.; Bai, X. Mask TextSpotter v3: Segmentation Proposal Network for Robust Scene Text Spotting. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; pp. 706–722.
7. Qiao, L.; Chen, Y.; Cheng, Z.; Xu, Y.; Niu, Y.; Pu, S.; Wu, F. Mango: A mask attention guided one-stage scene text spotter. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual, 2–9 February 2021; pp. 2467–2476.
8. Liu, Y.; Chen, H.; Shen, C.; He, T.; Jin, L.; Wang, L. ABCNet: Real-time scene text spotting with adaptive bezier-curve network. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Virtual, 14–19 June 2020; pp. 9809–9818.
9. Liu, X.; Liang, D.; Yan, S.; Chen, D.; Qiao, Y.; Yan, J. Fots: Fast oriented text spotting with a unified network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 5676–5685.
10. 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, Venice, Italy, 22–29 October 2017; pp. 2204–2212.
11. Cheng, M.; Sun, Y.; Wang, L.; Zhu, X.; Yao, K.; Chen, J.; Song, G.; Han, J.; Liu, J.; Ding, E.; et al. ViSTA: Vision and Scene Text Aggregation for Cross-Modal Retrieval. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 19–24 June 2022; pp. 5184–5193.
12. Liu, J.; Liu, X.; Sheng, J.; Liang, D.; Li, X.; Liu, Q. Pyramid mask text detector. *arXiv* **2019**, arXiv:1903.11800.
13. 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, Munich, Germany, 8–14 September 2018; pp. 67–83.
14. Huang, M.; Liu, Y.; Peng, Z.; Liu, C.; Lin, D.; Zhu, S.; Yuan, N.; Ding, K.; Jin, L. SwinTextSpotter: Scene Text Spotting via Better Synergy between Text Detection and Text Recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 19–24 June 2022; pp. 4593–4603.
15. Kittenplon, Y.; Lavi, I.; Fogel, S.; Bar, Y.; Manmatha, R.; Perona, P. Towards Weakly-Supervised Text Spotting using a Multi-Task Transformer. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 19–24 June 2022; pp. 4604–4613.
16. Long, S.; Qin, S.; Panteleev, D.; Bissacco, A.; Fujii, Y.; Raptis, M. Towards End-to-End Unified Scene Text Detection and Layout Analysis. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 19–24 June 2022; pp. 1049–1059.
17. Chen, J.; Yu, H.; Ma, J.; Li, B.; Xue, X. Text gestalt: Stroke-aware scene text image super-resolution. In Proceedings of the AAAI Conference on Artificial Intelligence, Arlington, VA, USA, 17–19 November 2022; pp. 285–293.
18. Peng, D.; Wang, X.; Liu, Y.; Zhang, J.; Huang, M.; Lai, S.; Li, J.; Zhu, S.; Lin, D.; Shen, C.; et al. SPTS: Single-Point Text Spotting. In Proceedings of the 30th ACM International Conference on Multimedia, Lisbon, Portugal, 10–14 October 2022; pp. 4272–4281.
19. 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. *arXiv* **2019**, arXiv:1908.08207.
20. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
21. Qiao, L.; Tang, S.; Cheng, Z.; Xu, Y.; Niu, Y.; Pu, S.; Wu, F. Text perceptron: Towards end-to-end arbitrary-shaped text spotting. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; pp. 11899–11907.
22. 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 Proceedings of the Asian Conference on Computer Vision, Perth, Australia, 2–6 December 2018; pp. 83–99.
23. Baek, Y.; Shin, S.; Baek, J.; Park, S.; Lee, J.; Nam, D.; Lee, H. Character region attention for text spotting. In Proceedings of the European Conference on Computer Vision, Virtual, 23–28 August 2020; pp. 504–521.

24. Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Virtual, 11–17 October 2021; pp. 10012–10022.
25. Xing, L.; Tian, Z.; Huang, W.; Scott, M. Convolutional character networks. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 9126–9136.
26. Liu, Y.; Shen, C.; Jin, L.; He, T.; Chen, P.; Liu, C.; Chen, H. Abcnet v2: Adaptive bezier-curve network for real-time end-to-end text spotting. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *44*, 8048–8064.
27. Wang, H.; Lu, P.; Zhang, H.; Yang, M.; Bai, X.; Xu, Y.; He, M.; Wang, Y.; Liu, W. All you need is boundary: Toward arbitrary-shaped text spotting. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; pp. 12160–12167.
28. Feng, W.; He, W.; Yin, F.; Zhang, X.; Liu, C. TextDragon: An end-to-end framework for arbitrary shaped text spotting. In Proceedings of the International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 9076–9085.
29. Wu, J.; Lyu, P.; Lu, G.; Zhang, C.; Yao, K.; Pei, W. Decoupling recognition from detection: Single shot self-reliant scene text spotter. In Proceedings of the 30th ACM International Conference on Multimedia, Lisbon, Portugal, 10–14 October 2022; pp. 1319–1328.
30. Zhong, H.; Tang, J.; Wang, W.; Yang, Z.; Yao, C.; Lu, T. Arts: Eliminating inconsistency between text detection and recognition with auto-rectification text spotter. *arXiv* **2021**, arXiv:2110.10405.
31. Lin, T.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2117–2125.
32. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969.
33. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 1137–1149. [CrossRef] [PubMed]
34. Kim, W.; Son, B.; Kim, I. Vilt: Vision-and-language transformer without convolution or region supervision. In Proceedings of the International Conference on Machine Learning, Virtual, 18–24 July 2021; pp. 5583–5594.
35. Zhao, L.; Wu, Z.; Wu, X.; Wilsbacher, G.; Wang, S. Background-Insensitive Scene Text Recognition with Text Semantic Segmentation. In Proceedings of the European Conference on Computer Vision, Tel Aviv, Israel, 23–27 October 2022; pp. 163–182.
36. Xu, X.; Zhang, Z.; Wang, Z.; Price, B.; Wang, Z.; Shi, H. Rethinking text segmentation: A novel dataset and a text-specific refinement approach. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Virtual, 19–25 June 2021; pp. 12045–12055.
37. Girshick, R. Fast r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 11–18 December 2015; pp. 1440–1448.
38. Yang, X.; He, D.; Zhou, Z.; Kifer, D.; Giles, C. Learning to Read Irregular Text with Attention Mechanisms. In Proceedings of the 26th International Joint Conference on Artificial Intelligence, Melbourne, Australia, 19–25 August 2017; p. 3.
39. He, T.; Tian, Z.; Huang, W.; Shen, C.; Qiao, Y.; Sun, C. An end-to-end textspotter with explicit alignment and attention. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 5020–5029.
40. Gupta, A.; Vedaldi, A.; Zisserman, A. Synthetic Data for Text Localisation in Natural Images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2315–2324.
41. Karatzas, D.; Shafait, F.; Uchida, S.; Iwamura, M.; Bigorda, L.; Mestre, S.; Mas, J.; Mota, D.; Almazan, J.; Heras, L.P.d. A ICDAR 2013 robust reading competition. In Proceedings of the 12th International Conference on Document Analysis and Recognition, Washington, DC, USA, 25–28 August 2013; pp. 1484–1493.
42. Karatzas, D.; Gomez-Bigorda, L.; Nicolaou, A.; Ghosh, S.; Bagdanov, A.; Iwamura, M.; Matas, J.; Neumann, L.; Chandrasekhar, V.; Lu, S. ICDAR 2015 competition on Robust Reading. In Proceedings of the 13th International Conference on Document Analysis and Recognition (ICDAR), Tunis, Tunisia, 23–26 August 2015; pp. 1156–1160.
43. Ch'ng, C.; Chan, C. Total-text: A comprehensive dataset for scene text detection and recognition. In Proceedings of the 14th IAPR International Conference on Document Analysis and Recognition, Kyoto, Japan, 9–15 November 2017; pp. 935–942.
44. Zhong, Z.; Jin, L.; Zhang, S.; Feng, Z. DeepText: A Unified Framework for Text Proposal Generation and Text Detection in Natural Images. *arXiv* **2016**, arXiv:1605.07314.
45. Veit, A.; Matera, T.; Neumann, L.; Matas, J.; Belongie, S. Coco-text: Dataset and benchmark for text detection and recognition in natural images. *arXiv* **2016**, arXiv:1601.07140.
46. Nayef, N.; Patel, Y.; Busta, M.; Chowdhury, P.; Karatzas, D.; Khlif, W.; Matas, J.; Pal, U.; Burie, J.; Liu, C.; et al. ICDAR2019 robust reading challenge on multi-lingual scene text detection and recognition—RRC-MLT-2019. In Proceedings of the 2019 International Conference on Document Analysis and Recognition (ICDAR), Sydney, Australia, 20–25 September 2019; pp. 1582–1587.
47. Liu, Y.; Jin, L.; Zhang, S.; Luo, C.; Zhang, S. Curved scene text detection via transverse and longitudinal sequence connection. *Pattern Recognit.* **2015**, *90*, 337–345. [CrossRef]
48. Ronen, R.; Tsiper, S.; Anschel, O.; Lavi, I.; Markovitz, A.; Manmatha, R. Glass: Global to local attention for scene-text spotting. In Proceedings of the 17th European Conference of Computer Vision, Tel Aviv, Israel, 24–28 October 2022; pp. 249–266.

49. 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, Munich, Germany, 8–14 September 2018; pp. 20–36.
50. Wang, W.; Xie, E.; Li, X.; Liu, X.; Liang, D.; Yang, Z.; Lu, T.; Shen, C. Pan++: Towards efficient and accurate end-to-end spotting of arbitrarily-shaped text. *IEEE Trans Pattern Anal. Mach. Intell.* **2021**, *44*, 5349–5367. [CrossRef] [PubMed]