EAFormer: Scene Text Segmentation with Edge-Aware Transformers

Haiyang Yu, Teng Fu, Bin Li*, and Xiangyang Xue

Shanghai Key Laboratory of Intelligent Information Processing
School of Computer Science, Fudan University
{hyyu20, libin, xyxue}@fudan.edu.cn, tfu23@m.fudan.edu.cn

Abstract. Scene text segmentation aims at cropping texts from scene images, which is usually used to help generative models edit or remove texts. The existing text segmentation methods tend to involve various text-related supervisions for better performance. However, most of them ignore the importance of text edges, which are significant for downstream applications. In this paper, we propose Edge-Aware Transformers, termed EAFormer, to segment texts more accurately, especially at the edge of texts. Specifically, we first design a text edge extractor to detect edges and filter out edges of non-text areas. Then, we propose an edge-guided encoder to make the model focus more on text edges. Finally, an MLP-based decoder is employed to predict text masks. We have conducted extensive experiments on commonly-used benchmarks to verify the effectiveness of EAFormer. The experimental results demonstrate that the proposed method can perform better than previous methods, especially on the segmentation of text edges. Considering that the annotations of several benchmarks (*e.g.*, COCO_TS and MLT_S) are not accurate enough to fairly evaluate our methods, we have relabeled these datasets. Through experiments, we observe that our method can achieve a higher performance improvement when more accurate annotations are used for training. The code and datasets are available at https://hyangyu.github.io/EAFormer/.

Keywords: Scene Text Segmentation · Edge-Guided Segmentation · Dataset Reannotation

1 Introduction

In the last decade, scene text segmentation [11, 32, 38, 54] has gained significant traction, primarily due to advancements in deep learning. The objective of a text segmentation model is to accurately distinguish between foregrounds (text areas) and backgrounds (non-text areas) at the pixel level. Scene text segmentation plays a crucial role in various applications, such as document analysis [15, 31], scene text image super-resolution [28, 35], scene understanding [13] and text

* Corresponding Author

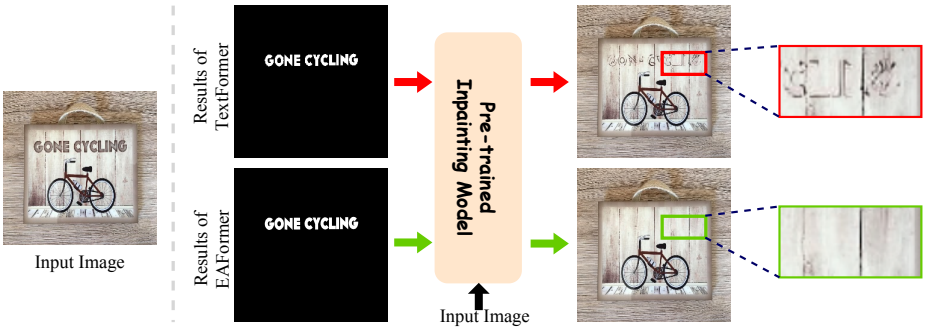

Fig. 1: Results comparison of the downstream application (text erasing) with different text masks as input. More accurate segmentation at text edges is beneficial to the text erasing task since less text pixels are wrongly predicted and more background information is reserved for the inpainting model.

erasing [10,12,24,50]. For example, TEAN [35] introduces the text segmentation results as auxiliary information to better super-resolve scene text images.

In order to promote the development of scene text segmentation, various methods [1, 41, 55] and datasets [3, 44, 45] have been proposed in recent years. Previous scene text segmentation methods tend to introduce text-related supervision, such as text or character recognition supervision, to improve performance. TexRNet [44] proposes a pre-trained character discriminator to introduce the supervision of character recognition, which requires additional annotations of character-level bounding boxes. Similarly, PGTSNet [45] designs a text perceptual discriminator to enhance the readability of segmentation results. In addition, both of them utilize various losses for better segmentation performance, which may make it challenging to select appropriate hyper-parameters to balance multiple losses. Recently, TextFormer [41] employs a recognition head to make the model focus on text details and improve its perception of texts. For the scene text segmentation task, there are several widely-used benchmarks, such as ICDAR13 FST [19], COCO_TS [3], MLT_S [4], Total-Text [8], TextSeg [44] and BTS [45]. Although the samples of these datasets seem to be adequate for deep-learning-based models, the annotation quality of some datasets (*e.g.*, MLT_S) may not meet the desired standards, especially in text-edge areas. The annotations of these datasets are obtained through bounding-box supervision, which cannot provide the same level of accuracy and precision as the datasets annotated by humans, such as TextSeg and BTS.

Although previous methods have achieved a certain performance improvement in text segmentation, they ignore the significance of text edges in practical applications. For instance, accurate text masks, especially in text-edge areas, can provide more background information to inpaint text areas in the text erasing task, as shown in Figure 1. In experiments, we observe that traditional edge detection algorithms, such as Canny [5], can well distinguish text edges. To fully exploit the merits of traditional edge detection methods to improve the

segmentation performance at text edges, in this paper, we propose Edge-Aware Transformers (EAFormer) for scene text segmentation. Specifically, EAFormer consists of three main modules: text edge extractor, edge-guided encoder and text segmentation decoder. The text edge extractor takes the scene images as input and predicts the text areas and edges. In this module, we adopt a light-weighted backbone to detect text areas and use the traditional edge detection algorithm Canny to obtain the edges of whole images. To alleviate the interference of edges in non-text areas, the masks of predicted text areas are used to filter out the edges of non-text areas. For the edge-guided encoder, we adopt the framework of SegFormer [43], which is composed of four stages. At the first stage of this encoder, we additionally design a symmetric cross-attention sub-module, which aims at utilizing the filtered text edges to guide the encoder to focus more on text edges at the first stage. Finally, the outputs of the edge-guided encoder are fused and input into the text segmentation decoder to predict text masks.

To validate the effectiveness of EAFormer, we have conducted extensive experiments on six text segmentation benchmarks. The results demonstrate that EAFormer can indeed improve the segmentation performance of the baseline model. However, as aforementioned, the annotations of COCO_TS and MLT_S are not accurate enough, which may make the experimental results on these two datasets unconvincing. In order to solve this problem, we have re-annotated the training, validation, and test sets of COCO_TS and MLT_S. Through experiments, we observe that the proposed EAFormer can still achieve better performance than previous methods with more accurate pixel-level annotations.

In summary, the contributions of this paper are as follows:

- For better segmentation performance in text-edge areas, we propose Edge-Aware Transformers (EAFormer) to explicitly predict text edges and use them to guide the following encoder.
- Considering the low-quality annotations of COCO_TS and MLT_S, we have re-annotated them for experiments to make the experimental results of EAFormer on these two datasets more convincing.
- Extensive experiments on six scene text segmentation benchmarks demonstrate that the proposed EAFormer can achieve state-of-the-art performance and perform better in text-edge areas.

2 Related Work

2.1 Scene Text Detection

Existing scene text detection methods can be divided into two categories: regression-based methods and segmentation-based methods. Regression-based methods [17, 18, 27, 53] regard text detection as a distinct object detection task, where the goal is to locate text regions by predicting the offsets from anchors or pixels. However, texts exhibit significant variations in scale and orientation compared to general objects. To handle oriented texts, EAST [53] directly regresses offsets

from boundaries in an anchor-free manner. While regression-based methods perform well for quadrilateral texts, they struggle to adapt to texts with arbitrary shapes. Segmentation-based methods [20, 25, 34, 37] consider text detection as a dense binary prediction task. DBNet [20] introduces differentiable binarization within a segmentation network, allowing for adaptive threshold prediction. Although various text detection methods have been proposed, we only employ a light-weighed backbone to detect text areas in our method. Although some methods [6,46] in general segmentation field have proposed to introduce the edge information to improve the performance, they are not perfectly suitable for the text segmentation task, which may result from two reasons: 1) To detect edges accurately, most of them need the annotations of edges, which is time-consuming and labor-intensive. 2) Directly employing them to solve text segmentation may introduce some edges of non-text areas, leading to subpar performance.

2.2 Semantic Segmentation

Semantic segmentation is a fundamental task in computer vision, which involves classifying each pixel in input images. Fully convolutional networks (FCN), which can learn dense prediction efficiently, were previously mainstream for the semantic segmentation task. To capture contextual information at multiple scales, several methods [7,51] introduce dilated convolutions or spatial pyramid pooling to enlarge the receptive field. Subsequently, attention mechanisms were introduced to better capture long-range dependencies [14, 47, 52]. Recently, a Transformer-based semantic segmentation method SegFormer [43] proposes to combine hierarchical Transformer encoders with a lightweight MLP decoder. Due to its outstanding performance, we adopt it as the baseline model of our method.

2.3 Scene Text Segmentation

Scene text segmentation aims at predicting fine-grained masks for texts in scene images. In the past, text segmentation methods often rely on thresholding [29,30, 33,36] or low-level features [2, 21, 39, 42] to binarize scene text images. However, these approaches often struggle with text images that have complex colors and textures, leading to poor performance. Recently, deep learning-based text segmentation methods have emerged. For instance, SMANet [3] adopts the encoder-decoder structure and introduces a new multi-scale attention module for scene text segmentation. TextFormer [41] introduces a text decoder into the hierarchical segmentation framework to enhance its ability to perceive text details. Due to the low labeling quality of previous datasets, TexRNet [44] proposes the TextSeg dataset with fine-grained annotations, which contain word- and character-level bounding polygons, masks, and transcriptions. Considering the lack of Chinese texts in text segmentation, a bilingual text segmentation dataset called BTS [45] has been proposed. The authors of BTS also developed PGTSNet, which employs a pre-trained text detection model to constrain text segmentation on detected text areas.

Fig. 2: Feature clustering results of PGTSNet and EAFormer. The visualization indicates that PGTSNet can hardly well perceive text edges compared with EAFormer.

3 Methodology

In this section, we introduce the proposed EAFormer in detail. First, we introduce the motivation of proposing EAFormer. Then, we detail each module of EAFormer, including text edge extractor, edge-guided encoder and text segmentation decoder. Finally, we introduce the loss function of our method.

3.1 Motivation

It is indisputable that text edges are crucial for the scene text segmentation task, especially for its downstream tasks like text erasing. Accurately segmenting text edges can provide more background information for the text erasing model to fill text areas. As shown in Figure 1, we utilize a pre-trained inpainting model, taking different types of text masks as input, to erase texts in scene images. Through experiments, we observe that the text bounding-box mask is too coarse to provide more backgrounds for the inpainting model. In addition, the text mask with inaccurate edge segmentation makes the inpainting model mistakenly regard the pixels belonging to texts as backgrounds, resulting in poor erasing results. Only when the text mask with accurate edge segmentation is provided, the inpainting model can generate satisfactory text erasing results.

Although PGTSNet [45] has realized the significance of text edges and employed a binary cross-entropy loss for detecting pixels at the text edges, it fails to explicitly introduce the easily available text edge information as one of the input information. In order to verify its ability to perceive text edges, we perform K-Means clustering on the features output by the backbone, where K is set to 3, representing the background, text edge, and text center, respectively. Through the visualization results shown in Figure 2, we observe that this method still has certain deficiencies in perceiving text edges.

In addition, we find that the traditional edge detection algorithm can obtain accurate text edges, which may benefit the scene text segmentation task. However, since the traditional edge detection method cannot distinguish text areas and non-text areas, most edges are detected in non-text areas. If the edge detection results are directly utilized as input to assist text segmentation, it could potentially confuse the text segmentation model and adversely impact its

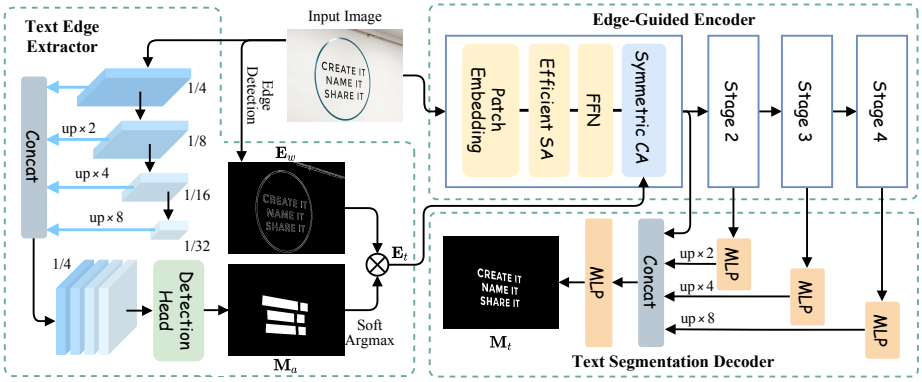

Fig. 3: Overall structure of EAFormer. EAFormer consists of three modules: text edge extractor, edge-guided encoder and text segmentation decoder. 'SA', 'CA', and 'FFN' represent self-attention, cross-attention, and feed-forward network, respectively.

performance. More discussions are in Section 5. In the following subsection, we will introduce how our method leverages the results of traditional edge detection algorithms to achieve better performance in the text segmentation task.

3.2 Edge-Aware Transformers (EAFormer)

As shown in Figure 3, the proposed EAFormer consists of three modules: text edge extractor, edge-guided encoder, and text segmentation decoder. Given the input scene text image $\mathbf{X} \in \mathbb{R}^{3\times H\times W}$, the text edge extractor is used to obtain the edges of text areas $\mathbf{E}_t$. Then, the text image $\mathbf{X}$ and detected text edges $\mathbf{E}_t$ are input into the edge-guided encoder to extract edge-aware features. Finally, the text segmentation decoder takes the features generated by the encoder as input to produce the corresponding text mask $\mathbf{M}_t$.

Text Edge Extractor. Since text edges are crucial for the scene text segmentation task, we propose a text edge extractor to obtain the edges of text areas. At first, we use the traditional edge detection algorithm Canny [5] to acquire the edges of the whole input image $\mathbf{E}_w$. As aforementioned, the edges of non-text areas in $\mathbf{E}_w$ may have negative impacts on text segmentation. Therefore, we introduce a lightweight text detection model in the text edge extractor to perform edge filtering. Specifically, we first use ResNet-like [16] backbone to extract multi-level visual features $\mathcal{F}^d = \{\mathbf{F}_1^d, \mathbf{F}_2^d, \mathbf{F}_3^d, \mathbf{F}_4^d\}$, where $\mathbf{F}_i^d \in \mathbb{R}^{C_i\times H_i\times W_i}$ represents the features at the i-th layer of the ResNet-like backbone (more details about the backbone of text detection are introduced in the supplementary material). Then, a text detection head is employed to predict the mask of text areas $\mathbf{M}_a$, which can be formulated as:

$$\mathbf{M}_a = \mathrm{Conv}_{1\times1}(\mathrm{Concat}(\{\mathbf{F}_1^d, \mathbf{F}_2^d, \mathbf{F}_3^d, \mathbf{F}_4^d\})) \tag{1}$$

where $\mathrm{Conv}_{1\times1}(\cdot)$ and $\mathrm{Concat}(\cdot)$ represent the 1×1 convolution layer and the concatenation operation, respectively. With the help of the mask of text areas

$\mathbf{M}_a$, we can filter out the edges of non-text areas through pixel-wise multiplication between the mask of text areas $\mathbf{M}_a$ and the detected edges $\mathbf{E}_w$. Therefore, the edges of text areas $\mathbf{E}_t$ can be obtained by:

$$\mathbf{E}_t = \mathbf{M}_a \odot \text{SoftArgmax}(\mathbf{E}_w) \tag{2}$$

It is worth mentioning that we exert the soft argmax operation on $\mathbf{E}_w$ before multiplication since joint optimizing the text detection and segmentation branch can achieve better text detection performance. Then, the filtered text edges $\mathbf{E}_t$ are subsequently input into the following edge-guided encoder to enhance its ability to distinguish pixels around text edges.

Edge-Guided Encoder. Since SegFormer [43] exhibits outstanding abilities in semantic segmentation, we adopt it as the basic framework of the edge-guided encoder. As shown in Figure 3, the edge-guided encoder consists of four stages, and the filtered text edges are merged at the first stage. Each encoding stage contains three submodules: overlap patch embedding, efficient self-attention, and feed-forward network. The overlap patch embedding is used to extract local features around each patch. Then, the features are input into the self-attention layer to excavate the correlations between pixels. The vanilla self-attention layer is formulated as follows:

$$\text{SA}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Softmax}(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_{\text{head}}}})\mathbf{V}, \tag{3}$$

where $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ are obtained by applying different embedding layers to the same features. To reduce the computational cost, we follow [43] to introduce the spatial reduction operation for $\mathbf{K}$ and $\mathbf{V}$. More details of spatial reduction are shown in the supplementary material. Finally, for the i-th stage, a feed-forward network is employed to generate the output features $\mathbf{F}_i^s$. Differently, we additionally introduce a symmetric cross-attention layer after the feed-forward network of the first stage to merge the extracted edge guidance $\mathbf{E}_t$. Specifically, the symmetric cross-attention layer includes two cross-attention operations between the features $\mathbf{F}_1^s$ from the first stage and the edge guidance $\mathbf{E}_t$. On one hand, $\mathbf{E}_t$ is regarded as Query to extract the edge-aware visual information $\mathbf{F}^{ev}$, where $\mathbf{F}_1^s$ is viewed as Key and Value; on the other hand, $\mathbf{F}_1^s$ is used as Query to further excavate the useful text edge information $\mathbf{F}^{te}$, where $\mathbf{E}_t$ is seen as Key and Value. Therefore, the final output of the first stage $\hat{\mathbf{F}}_1^s$ can be expressed as:

$$\begin{aligned} \hat{\mathbf{F}}_1^s &= \mathbf{F}^{ev} \oplus \mathbf{F}^{te} \oplus \mathbf{F}_1^s \\ \mathbf{F}^{ev} &= \text{SA}(\mathbf{E}_t, \mathbf{F}_1^s, \mathbf{F}_1^s) \\ \mathbf{F}^{te} &= \text{SA}(\mathbf{F}_1^s, \mathbf{E}_t, \mathbf{E}_t) \end{aligned} \tag{4}$$

where $\text{SA}(\cdot)$ represents the aforementioned self-attention operation, $\oplus$ denotes the pixel-wise addition. Subsequently, $\hat{\mathbf{F}}_1^s$ and outputs of other stages are input into the text segmentation decoder.

Text Segmentation Decoder. Similar to the previous method [41], we employ several MLP layers to fuse features and predict the final text masks $\mathbf{M}_t$. First, we

unify the channel dimension of the outputs of four stages through corresponding MLP layers. Then, they are up-sampled into the same resolution and further fused by an MLP layer. Finally, the fused features are used to predict the text masks. Assuming that the resolution of features from the i-th stage is $H_i \times W_i \times C_i$, the decoding process can be formulated as:

$$\begin{aligned}
\tilde{\mathbf{F}}_i^s &= \mathrm{MLP}(C_i, C_1)(\mathbf{F}),\ \mathbf{F} \in \{\hat{\mathbf{F}}_1^s, \mathbf{F}_2^s, \mathbf{F}_3^s, \mathbf{F}_4^s\} \\
\tilde{\mathbf{F}}_i^s &= \mathrm{UpSample}(H_1, W_1)(\tilde{\mathbf{F}}_i^s),\ i \in \{1, 2, 3, 4\} \\
\mathbf{F}^s &= \mathrm{Fuse}(\mathcal{F}),\ \mathcal{F} = \{\tilde{\mathbf{F}}_1^s, \tilde{\mathbf{F}}_2^s, \tilde{\mathbf{F}}_3^s, \tilde{\mathbf{F}}_4^s\} \\
\mathbf{M}_t &= \mathrm{MLP}(C_1, 2)(\mathbf{F}^s)
\end{aligned} \tag{5}$$

where $\mathrm{MLP}(C_{\text{in}}, C_{\text{out}})(\cdot)$ represents that the channels of input and output features in MLP are C_{in} and C_{out}, respectively. $\mathrm{Fuse}(\cdot)$ denotes that the input features are first concatenated and then reduced in the channel dimension through an MLP layer.

3.3 Loss Function

Previous text segmentation methods [44, 45] tend to introduce various losses to improve the performance, which may bring difficulties in choosing appropriate hyper-parameters. In the proposed EAFormer, only two cross-entropy losses, text detection loss $\mathcal{L}_{det}$ and text segmentation loss $\mathcal{L}_{seg}$, are used for optimization, which can be expressed by:

$$\mathcal{L} = \underbrace{\mathrm{CE}(\mathbf{M}_t, \hat{\mathbf{M}}_t)}_{\mathcal{L}_{seg}} + \lambda\ \underbrace{\mathrm{CE}(\mathbf{M}_a, \hat{\mathbf{M}}_a)}_{\mathcal{L}_{det}} \tag{6}$$

where λ is the hyper-parameter to balance $\mathcal{L}_{det}$ and $\mathcal{L}_{seg}$; $\hat{\mathbf{M}}_a$ and $\hat{\mathbf{M}}_t$ are the ground truth of $\mathbf{M}_a$ and $\mathbf{M}_t$, respectively. Please note that the utilized bounding box-level supervision for $\mathbf{M}_a$ can be obtained from the semantic-level annotations, which means that the proposed method only requires the semantic-level annotations as the same as previous methods.

4 Experiments

4.1 Datasets

In this paper, we have conducted extensive experiments on six text segmentation benchmarks, including five English text segmentation datasets (ICDAR13 FST [19], COCO_TS [3], MLT_S [4], Total-Text [8], and TextSeg [44]) and one bi-lingual text segmentation dataset BTS [45]. Some statistical details about each dataset are shown in Table 1. The examples of each dataset are displayed in the supplementary material.

As shown in Figure 4, the original annotations of COCO_TS and MLT_S are too coarse to train a text segmentation model with satisfactory performance.

Dataset	Images	Train/Test/Valid	Words	Classes	Languages
ICDAR13 FST [19]	462	229/233/-	1,944	36	English
COCO_TS [3]	14,690	11,882/2,808/-	139,034	36	English
MLT_S [4]	6,896	5,540/1,356/-	30,691	36	English
Total-Text [8]	1,555	1,255/300/-	9,330	36	English
TextSeg [44]	4,024	2,646/1,038/340	15,691	36	English
BTS [45]	14,287	10,191/1,366/2,730	44,385	3,988	Chinese,English

Table 1: Statistical details about the adopted six text segmentation datasets.

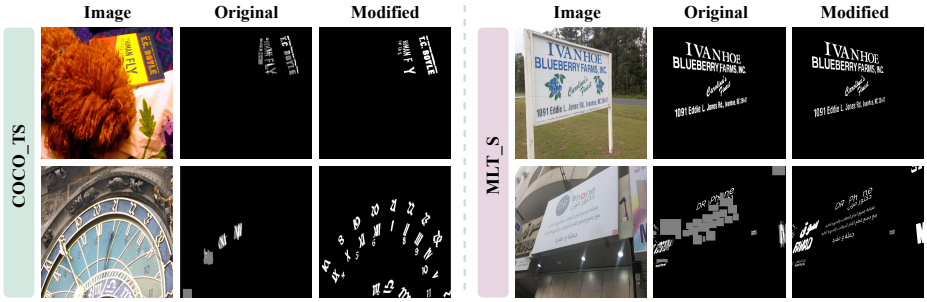

Fig. 4: Comparison between original and modified annotations. The original datasets have the problems of missing and inaccurate annotations. Using re-annotated datasets to train the proposed method makes the experimental results more convincing.

Even if the proposed method achieves better performance on these datasets, it is not sufficient to demonstrate the effectiveness of our method. To make the experimental results more convincing, we have re-annotated all samples of these two datasets and use the newly annotated datasets to conduct experiments. Figure 4 shows the comparison between original and modified annotations.

4.2 Implementation Details

The proposed method is implemented with PyTorch, and we conduct all experiments on 8 NVIDIA RTX 4090 GPUs. The AdamW [23] optimizer is adopted with an initial learning rate 6×10^{-5} in all experiments, and the weight decay is set to 0.01. The batch size is set to 4. Following the previous methods [32,41,45], we also adopt some data augmentation operations, such as random cropping and flipping, in the training phase. Different from existing methods employing pre-trained models to detect text areas [45] or recognize characters [44], all modules in the proposed EAFormer are jointly trained. In other words, no additional datasets are used when training EAFormer. Two thresholds of Canny are set to 100 and 200, respectively. To evaluate the performance of the proposed method, we utilize both foreground Intersection-over-Union (fgIoU) and F-score on foreground pixels. The metrics of fgIoU and F-score follow the percentage format and decimal format, respectively.

Method	ICDAR13 FST		COCO_TS		MLT_S		Total-Text		TextSeg	
	fgIoU	F-score	fgIoU	F-score	fgIoU	F-score	fgIoU	F-score	fgIoU	F-score
PSPNet [3,51]	-	0.797	-	-	-	-	-	0.740	-	-
SMANet [4]	-	0.785	-	-	-	-	-	0.770	-	-
DeeplabV3+ [7]	69.27	0.802	72.07	0.641	84.63	0.837	74.44	0.824	84.07	0.914
HRNetV2-W48 [40]	70.98	0.822	68.93	0.629	83.26	0.836	75.29	0.825	85.03	0.914
HRNet-OCR [49]	72.45	0.830	69.54	0.627	83.49	0.838	76.23	0.832	85.98	0.918
TexRNet [44]	**73.38**	0.850	72.39	0.720	86.09	0.865	78.47	0.848	86.84	0.924
ARM-Net [32]	-	**0.851**	-	-	-	-	-	0.854	-	0.927
TFT [48]	<u>72.71</u>	<u>0.845</u>	<u>73.40</u>	<u>0.847</u>	<u>87.80</u>	<u>0.935</u>	<u>82.10</u>	<u>0.902</u>	87.11	0.931
TextFormer [41]	72.27	0.838	73.20	0.745	86.66	0.908	81.56	0.887	<u>87.42</u>	<u>0.933</u>
SegFormer (Base) [43]	60.44	0.753	63.17	0.774	78.77	0.863	73.31	0.846	84.59	0.916
EAFormer (Ours)	72.63	0.840	**81.03**	**0.895**	**89.02**	**0.942**	**82.73**	**0.906**	**88.06**	**0.939**

Table 2: Performance comparison on five English text segmentation datasets. The bold and underlined numbers represent the best and second-best results, respectively.

Method	COCO_TS		MLT_S	
	fgIoU	F-score	fgIoU	F-score
TextFormer [41]	52.73	0.688	74.83	0.861
EAFormer (Ours)	**64.82**	**0.786**	**81.92**	**0.900**

Table 3: The experimental results on re-annotated COCO_TS and MLT_S. The results of TextFormer are obtained by re-implementing the method.

4.3 Experimental Results

Quantitative Comparison. To comprehensively evaluate EAFormer, we have conducted experiments on both English and bi-lingual text segmentation datasets. The experimental results on five English text segmentation datasets are shown in Table 2. Compared with previous methods, EAFormer can achieve a clear improvement in fgIoU and F-score on most benchmarks. For example, on TextSeg, EAFormer outperforms the previous SOTA method TextFormer [41] by 0.64% and 0.6% in fgIoU and F-score, respectively. Although the original COCO_TS and MLT_S datasets have coarse annotations, the proposed EAFormer can still exhibit better performance, such as achieving a 7.63% improvement in fgIoU on the COCO_TS dataset compared with TFT [48]. Considering that the experimental results based on inaccurate annotations are not convincing enough, we have re-annotated both the training dataset and the test dataset of COCO_TS and MLT_S. The experimental results based on re-annotated datasets are displayed in Table 3. Through experiments, we observe that the proposed EAFormer can still achieve a considerable performance improvement when the datasets with more accurate annotations are used for training and test. Compared with the results on original datasets, the performance on re-annotated datasets seems to drop a lot. The following two reasons may explain this phenomenon: 1) There are many blurred texts in the datasets, which indeed bring certain challenges

Method	BTS	
	fgIoU	F-score
DeeplabV3+ [7]	71.15	0.796
HRNetV2-W48 [40]	81.84	0.861
HRNetV2-W48+OCR [49]	82.76	0.866
TexRNet (DeeplabV3+, w/o Cls) [44]	83.68	0.883
TexRNet (DeeplabV3+, w/ Cls) [44]	83.81	0.892
PGTSNet [45]	86.48	0.909
TFT [48]	<u>87.84</u>	<u>0.935</u>
TextFormer [41]	86.97	0.930
SegFormer (Baseline) [43]	84.99	0.908
EAFormer (Ours)	**88.08**	**0.937**

Table 4: Performance comparison on BTS. The bold and underlined numbers represent the best and second-best results, respectively.

to the model to deal with text edges; 2) The re-annotated test datasets are more accurate and there are no ignored areas in evaluation. In addition, we also conduct experiments on the bi-lingual text segmentation dataset BTS [45], and the results are shown in Table 4. Although PGTSNet unfairly introduces a pre-trained text detector, EAFormer can still achieve an improvement of 1.6%/2.8% in fgIoU/F-score, which validates the effectiveness of the proposed method.

Since we introduce a light-weighted text detection head, it is inevitable to introduce more parameters. We have evaluated the number of parameters and inference speed. Compared with the previous SOTA method TextFormer (85M parameters and 0.42s per image), the proposed model has 92M parameters and costs 0.47s per image in average. With slight increasing in the number of parameters, our method can achieve significant performance improvements.

Qualitative Comparison. We also compare EAFormer with previous methods in term of segmentation quality through visualizations. As shown in Figure 5, the proposed EAFormer can perform better than previous methods at text edges, which benefits from the introduced edge information. In addition, for COCO_TS and MLT_S, we compare the segmentation results based on both the original and modified annotations. Although Table 3 indicates that the performance of our method has declined when using the re-annotated datasets for training and test, the visualizations in Figure 5 demonstrate that our model is able to achieve better segmentation results based on the re-annotated datasets. More visualizations are shown in the supplementary material.

4.4 Ablation Study

Hyper-parameter λ. During training EAFormer, two losses are used for optimization. The hyper-parameter λ is to balance the weights of $\mathcal{L}_{det}$ and $\mathcal{L}_{seg}$, and an appropriate λ may result in better performance. Thus, we conduct some experiments to select λ ranging from $\{0.1, 0.5, 1.0, 5.0, 10.0\}$, and the experimental

λ	TextSeg		BTS	
	fgIoU	F-score	fgIoU	F-score
0.1	84.03	0.910	86.45	0.913
0.5	87.33	0.926	87.03	0.931
1.0	**88.06**	**0.939**	**88.08**	**0.937**
5.0	87.67	0.934	87.62	0.935
10.0	87.94	0.937	87.58	0.933

Table 5: The experimental results of choosing λ. When λ is set to 1.0, the proposed method can achieve the best performance.

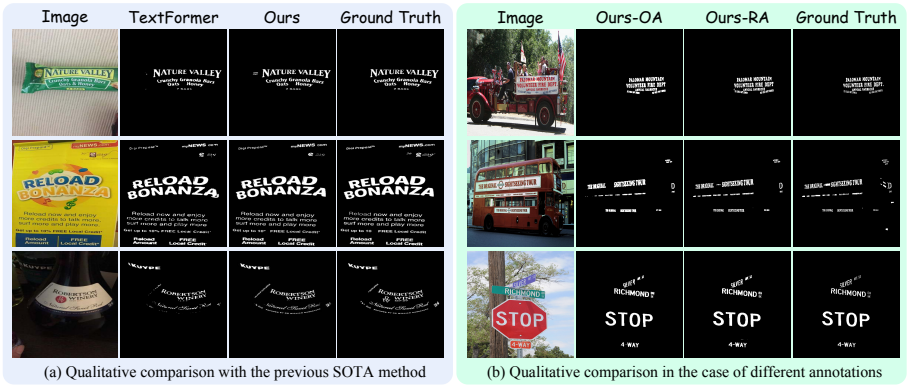

Fig. 5: Visualizations of qualitative comparison between different methods or training with different annotations. 'OA' and 'RA' indicate training EAFormer with original annotations and re-annotations, respectively.

results are shown in Table 5. When λ is set to 1.0, EAFormer reaches the best and it boosts fgIoU/F-score by 3.47%/2.3% compared with the baseline model on TextSeg. The results shown in Table 5 indicate that λ has little impact on performance when it ranges from $\{0.5, 1.0, 5.0, 10.0\}$. However, if λ is set to 0.1, the performance of EAFormer is unsatisfactory, which may result from that too small λ makes it difficult for the text detection module to converge and further affects the performance of text segmentation. Therefore, we set λ to 1.0 for all experiments in this paper.

Edge Filtering and Edge Guidance. In the proposed EAFormer, the edge filtering in the text edge extractor and the edge guidance in the edge-guided encoder are two key components. To evaluate the performance gain of these two strategies, we conduct ablation experiments on them and the results are shown in Table 6. Please note that when only edge filtering is used, the extracted edge information is concatenated with the input image and fed into the SegFormer-based encoder. As shown Table 6, introducing edge filtering can lead to a clear performance improvement. However, if only edge guidance is introduced, the

EF	EG	TextSeg		BTS	
		fgIoU	F-score	fgIoU	F-score
		84.59	0.916	84.99	0.908
✓		86.85	0.927	87.35	0.922
	✓	81.03	0.832	80.35	0.828
✓	✓	88.06	0.939	88.08	0.937

Table 6: The experimental results of ablation studies. If only the edge guidance is introduced, the edges of non-text areas have negative impacts on the proposed method.

performance of our method is subpar. A possible reason is that the edges of non-text areas introduce more interference information, resulting in that the model cannot effectively use the extracted edges to assist text segmentation. Therefore, both edge filtering and edge guidance are necessary for our method, and when both of them are adopted, EAFormer can achieve the SOTA performance.

5 Discussions

Filtering out the edges of non-text areas. In the text edge extractor module, we propose to filter out edge information in non-text areas to avoid their adverse impact on model performance. In the section of the ablation experiment, we can know that filtering the edge information of non-text areas can clearly improve the performance. Through visualizations (see the supplementary material), we observe that when all edge information is used to assist segmentation, the model will mistakenly believe that areas with edge information should be classified as foreground. Therefore, in order to give the model explicit edge guidance, the proposed method only retains the edge information of the text area as input.
Introducing text edges at different layers. In the edge-guided encoder, we extract edge-enhanced feature information only in the first stage through symmetric cross-attention. It is well known that lower-level features are more sensitive to text edge information. We have visualized the clustering results of features at different stages in Figure 6, and the visualizations indicate that only the features of the first stage focus on the edge information. Therefore, it is reasonable and effective to introduce detected edges at an earlier stage. We also tried to introduce the edge guidance at other stages to conduct experiments (detailed results are shown in the supplementary material). The experimental results indicate that the higher the stage at which the detected edges are introduced, the smaller the performance improvement of EAFormer. In particular, when the detected edges are introduced at the third or fourth stage, the performance of EAFormer is even lower than the baseline.
Utilizing an off-the-shelf text detector. In the text edge extractor, we employ a lightweight text detector consisting of a ResNet-based backbone and an MLP decoder. In fact, we can utilize an off-the-shelf text detector that has been pre-trained on text detection datasets, which can help EAFormer achieve better

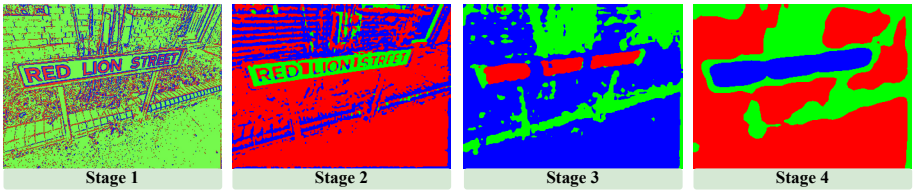

Fig. 6: The clustering results of features from different stages. It is reasonable to introduce the text-edge guidance at the first stage since only the features of the first stage focus on edge information.

performance in practical applications. Since this may be unfair to the previous method, we only explore the performance upper limit of EAFormer. In experiments, using pre-trained DBNet [20] to replace the lightweight text detector module, the performance of EAFormer on TextSeg can reach a new SOTA performance (90.16%/95.2% in fgIoU/F-score).

Differences from previous edge-guided methods. Actually, the incorporation of edge information for segmentation is a well-explored strategy [9, 22, 26]. However, our method still has some differences from previous works. First, BCANet [26] and BSNet [9] need edge supervision while the proposed method directly employs Canny to extract edges. Although EGCAN [22] also uses Canny, our method additionally introduce edge filtering to reserve useful edge information, which is specifically designed for text segmentation. In addition, EGCAN fuses the edge information in all encoder layers while our method fuses the edge information only in the first layer through our designed symmetric cross-attention.

Limitations. To filter the edges of non-text areas, we introduce a light-weighted text detector, which may slightly increase the number of learnable parameters. In addition, we only utilize the off-the-shelf edge detection algorithm Canny to extract text edges rather than employing a better deep-learning-based edge detection method. Introducing a SOTA edge detection method may further improve the performance of our method.

6 Conclusion

In this paper, we propose Edge-Aware Transformers, called EAFormer, to solve inaccurate text segmentation at text edges. Specifically, the traditional edge detection algorithm Canny is employed to extract edges. To avoid involving the edges of non-text areas, a light-weighted text detection module is adopted to filter out the useless edges for segmenting texts. In addition, based on SegFormer, we propose an edge-guided encoder to enhance its ability to perceive text edges. Considering that the low-quality annotations of several datasets may affect the credibility of experimental results, we have re-annotated these datasets. Extensive experiments are conducted on publicly available benchmarks, and the SOTA results validate the effectiveness of EAFormer in the text segmentation task.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China (No.62176060), STCSM project (No.22511105000), Shanghai Municipal Science and Technology Major Project (No.2021SHZDZX0103), and the Program for Professor of Special Appointment (Eastern Scholar) at Shanghai Institutions of Higher Learning.

References

1. Andreini, P., Ciano, G., Bonechi, S., Graziani, C., Lachi, V., Mecocci, A., Sodi, A., Scarselli, F., Bianchini, M.: A two-stage gan for high-resolution retinal image generation and segmentation. Electronics **11**(1), 60 (2021)
2. Bai, B., Yin, F., Liu, C.L.: A seed-based segmentation method for scene text extraction. In: 2014 11th IAPR International Workshop on Document Analysis Systems. pp. 262–266. IEEE (2014)
3. Bonechi, S., Andreini, P., Bianchini, M., Scarselli, F.: Coco_ts dataset: pixel–level annotations based on weak supervision for scene text segmentation. In: International Conference on Artificial Neural Networks. pp. 238–250. Springer (2019)
4. Bonechi, S., Bianchini, M., Scarselli, F., Andreini, P.: Weak supervision for generating pixel–level annotations in scene text segmentation. Pattern Recognition Letters **138**, 1–7 (2020)
5. Canny, J.: A computational approach to edge detection. IEEE Transactions on pattern analysis and machine intelligence (6), 679–698 (1986)
6. Chen, J., Li, J., Pan, D., Zhu, Q., Mao, Z.: Edge-guided multiscale segmentation of satellite multispectral imagery. IEEE Transactions on Geoscience and Remote Sensing **50**(11), 4513–4520 (2012)
7. Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: Proceedings of the European conference on computer vision (ECCV). pp. 801–818 (2018)
8. Ch'ng, C.K., Chan, C.S.: Total-text: A comprehensive dataset for scene text detection and recognition. In: 2017 14th IAPR international conference on document analysis and recognition (ICDAR). vol. 1, pp. 935–942. IEEE (2017)
9. Cong, R., Zhang, Y., Yang, N., Li, H., Zhang, X., Li, R., Chen, Z., Zhao, Y., Kwong, S.: Boundary guided semantic learning for real-time covid-19 lung infection segmentation system. IEEE Transactions on Consumer Electronics **68**(4), 376–386 (2022)
10. Conrad, B., Chen, P.I.: Two-stage seamless text erasing on real-world scene images. In: 2021 IEEE International Conference on Image Processing (ICIP). pp. 1309–1313. IEEE (2021)
11. Dai, Y., Huang, Z., Gao, Y., Xu, Y., Chen, K., Guo, J., Qiu, W.: Fused text segmentation networks for multi-oriented scene text detection. In: 2018 24th international conference on pattern recognition (ICPR). pp. 3604–3609. IEEE (2018)
12. Du, X., Zhou, Z., Zheng, Y., Ma, T., Wu, X., Jin, C.: Modeling stroke mask for end-to-end text erasing. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 6151–6159 (2023)
13. Ess, A., Müller, T., Grabner, H., Van Gool, L.: Segmentation-based urban traffic scene understanding. In: BMVC. vol. 1, p. 2. Citeseer (2009)

14. Fu, J., Liu, J., Jiang, J., Li, Y., Bao, Y., Lu, H.: Scene segmentation with dual relation-aware attention network. IEEE Transactions on Neural Networks and Learning Systems **32**(6), 2547–2560 (2020)
15. Fujisawa, H., Nakano, Y., Kurino, K.: Segmentation methods for character recognition: from segmentation to document structure analysis. Proceedings of the IEEE **80**(7), 1079–1092 (1992)
16. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
17. He, W., Zhang, X.Y., Yin, F., Liu, C.L.: Deep direct regression for multi-oriented scene text detection. In: Proceedings of the IEEE international conference on computer vision. pp. 745–753 (2017)
18. He, W., Zhang, X.Y., Yin, F., Liu, C.L.: Multi-oriented and multi-lingual scene text detection with direct regression. IEEE Transactions on Image Processing **27**(11), 5406–5419 (2018)
19. 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: 2013 12th international conference on document analysis and recognition. pp. 1484–1493. IEEE (2013)
20. Liao, M., Wan, Z., Yao, C., Chen, K., Bai, X.: Real-time scene text detection with differentiable binarization. In: Proceedings of the AAAI conference on artificial intelligence. vol. 34, pp. 11474–11481 (2020)
21. Liu, X., Samarabandu, J.: Multiscale edge-based text extraction from complex images. In: 2006 IEEE International Conference on Multimedia and Expo. pp. 1721–1724. IEEE (2006)
22. Liu, Z., Li, J., Song, R., Wu, C., Liu, W., Li, Z., Li, Y.: Edge guided context aggregation network for semantic segmentation of remote sensing imagery. Remote Sensing **14**(6), 1353 (2022)
23. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: International Conference on Learning Representations (2018)
24. Lyu, G., Liu, K., Zhu, A., Uchida, S., Iwana, B.K.: Fetnet: Feature erasing and transferring network for scene text removal. Pattern Recognition **140**, 109531 (2023)
25. Lyu, P., Yao, C., Wu, W., Yan, S., Bai, X.: Multi-oriented scene text detection via corner localization and region segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 7553–7563 (2018)
26. Ma, H., Yang, H., Huang, D.: Boundary guided context aggregation for semantic segmentation. arXiv preprint arXiv:2110.14587 (2021)
27. Ma, J., Shao, W., Ye, H., Wang, L., Wang, H., Zheng, Y., Xue, X.: Arbitrary-oriented scene text detection via rotation proposals. IEEE transactions on multimedia **20**(11), 3111–3122 (2018)
28. Ma, J., Jin, L., Zhang, J., Jiang, J., Xue, Y., He, M.: Textsrnet: Scene text super-resolution based on contour prior and atrous convolution. In: 2022 26th International Conference on Pattern Recognition (ICPR). pp. 3252–3258. IEEE (2022)
29. Mustafa, W.A., Kader, M.M.M.A.: Binarization of document image using optimum threshold modification. In: Journal of Physics: Conference Series. vol. 1019, p. 012022. IOP Publishing (2018)
30. Otsu, N.: A threshold selection method from gray-level histograms. IEEE transactions on systems, man, and cybernetics **9**(1), 62–66 (1979)

31. Pack, C., Soh, L.K., Lorang, E.: Perceptual cue-guided adaptive image downscaling for enhanced semantic segmentation on large document images. International Journal on Document Analysis and Recognition (IJDAR) pp. 1–17 (2023)
32. Ren, Y., Zhang, J., Chen, B., Zhang, X., Jin, L.: Looking from a higher-level perspective: Attention and recognition enhanced multi-scale scene text segmentation. In: Proceedings of the Asian Conference on Computer Vision. pp. 3138–3154 (2022)
33. Sauvola, J., Pietikäinen, M.: Adaptive document image binarization. Pattern recognition **33**(2), 225–236 (2000)
34. Shi, B., Bai, X., Belongie, S.: Detecting oriented text in natural images by linking segments. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2550–2558 (2017)
35. Shu, R., Zhao, C., Feng, S., Zhu, L., Miao, D.: Text-enhanced scene image super-resolution via stroke mask and orthogonal attention. IEEE Transactions on Circuits and Systems for Video Technology (2023)
36. Su, B., Lu, S., Tan, C.L.: Binarization of historical document images using the local maximum and minimum. In: Proceedings of the 9th IAPR International Workshop on Document Analysis Systems. pp. 159–166 (2010)
37. Tang, J., Yang, Z., Wang, Y., Zheng, Q., Xu, Y., Bai, X.: Seglink++: Detecting dense and arbitrary-shaped scene text by instance-aware component grouping. Pattern recognition **96**, 106954 (2019)
38. Tang, Y., Wu, X.: Scene text detection and segmentation based on cascaded convolution neural networks. IEEE transactions on Image Processing **26**(3), 1509–1520 (2017)
39. Vo, Q.N., Kim, S.H., Yang, H.J., Lee, G.: Binarization of degraded document images based on hierarchical deep supervised network. Pattern Recognition **74**, 568–586 (2018)
40. Wang, J., Sun, K., Cheng, T., Jiang, B., Deng, C., Zhao, Y., Liu, D., Mu, Y., Tan, M., Wang, X., et al.: Deep high-resolution representation learning for visual recognition. IEEE transactions on pattern analysis and machine intelligence **43**(10), 3349–3364 (2020)
41. Wang, X., Wu, C., Yu, H., Li, B., Xue, X.: Textformer: Component-aware text segmentation with transformer. In: 2023 IEEE International Conference on Multimedia and Expo (ICME). pp. 1877–1882. IEEE (2023)
42. Wu, Y., Natarajan, P., Rawls, S., AbdAlmageed, W.: Learning document image binarization from data. In: 2016 IEEE International Conference on Image Processing (ICIP). pp. 3763–3767. IEEE (2016)
43. Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J.M., Luo, P.: Segformer: Simple and efficient design for semantic segmentation with transformers. Advances in Neural Information Processing Systems **34**, 12077–12090 (2021)
44. 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. pp. 12045–12055 (2021)
45. Xu, X., Qi, Z., Ma, J., Zhang, H., Shan, Y., Qie, X.: Bts: a bi-lingual benchmark for text segmentation in the wild. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 19152–19162 (2022)
46. Yin, X., Li, X., Ni, P., Xu, Q., Kong, D.: A novel real-time edge-guided lidar semantic segmentation network for unstructured environments. Remote Sensing **15**(4), 1093 (2023)

47. Yu, C., Wang, J., Gao, C., Yu, G., Shen, C., Sang, N.: Context prior for scene segmentation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 12416–12425 (2020)
48. Yu, H., Wang, X., Niu, K., Li, B., Xue, X.: Scene text segmentation with text-focused transformers. In: Proceedings of the 31st ACM International Conference on Multimedia. pp. 2898–2907 (2023)
49. Yuan, Y., Chen, X., Wang, J.: Object-contextual representations for semantic segmentation. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16. pp. 173–190. Springer (2020)
50. Zdenek, J., Nakayama, H.: Erasing scene text with weak supervision. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 2238–2246 (2020)
51. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid scene parsing network. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2881–2890 (2017)
52. Zheng, S., Lu, J., Zhao, H., Zhu, X., Luo, Z., Wang, Y., Fu, Y., Feng, J., Xiang, T., Torr, P.H., et al.: Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 6881–6890 (2021)
53. Zhou, X., Yao, C., Wen, H., Wang, Y., Zhou, S., He, W., Liang, J.: East: an efficient and accurate scene text detector. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 5551–5560 (2017)
54. Zhou, Y., Feild, J., Learned-Miller, E., Wang, R.: Scene text segmentation via inverse rendering. In: 2013 12th International Conference on Document Analysis and Recognition. pp. 457–461. IEEE (2013)
55. Zu, X., Yu, H., Li, B., Xue, X.: Weakly-supervised text instance segmentation. In: Proceedings of the 31st ACM International Conference on Multimedia. pp. 1915–1923 (2023)