STIRER: A Unified Model for Low-Resolution Scene Text Image Recovery and Recognition

Minyi Zhao
zhaomy20@fudan.edu.cn
School of Computer Science
Fudan University
Shanghai, China

Shijie Xuyang
ysjxu22@m.fudan.edu.cn
School of Computer Science
Fudan University
Shanghai, China

Jihong Guan
jhguan@tongji.edu.cn
Department of Computer Science and Technology
Tongji University
Shanghai, China

Shuigeng Zhou*
sgzhou@fudan.edu.cn
School of Computer Science
Fudan University
Shanghai, China

ABSTRACT

Though scene text recognition (STR) from high-resolution (HR) images has achieved significant success in the past years, text recognition from low-resolution (LR) images is still a challenging task. This inspires the study on scene text image super-resolution (STISR) to generate super-resolution (SR) images based on the LR images, then STR is performed on the generated SR images, which eventually boosts the recognition performance. However, existing methods have two major drawbacks: 1) STISR models may generate imperfect SR images, which mislead the subsequent recognition. 2) As the STISR models are optimized for high recognition accuracy, the fidelity of SR images may be degraded. Consequently, neither the recognition performance of STR nor the fidelity of STISR is desirable. In this paper, a novel model called **STIRER** (the abbreviation of **S**cene **T**ext **I**mage **RE**covery and **R**ecognition) is proposed to effectively and simultaneously recover and recognize LR scene text images under a unified framework. Concretely, STIRER consists of a feature encoder to obtain pixel features and two dedicated decoders to generate SR images and recognize texts respectively based on the encoded features and the raw LR images. We propose a progressive scene text swin transformer architecture as the encoder to enrich the representations of the pixel features for better recovery and recognition. Extensive experiments on two LR datasets show the superiority of our model to the existing methods on recognition performance, super-resolution fidelity and computational cost. The STIRER Code is available in https://github.com/zhaominyiz/STIRER.

*Corresponding author.

MM '23, October 29-November 3, 2023, Ottawa, ON, Canada

ACM ISBN 979-8-4007-0108-5/23/10…$15.00
https://doi.org/10.1145/3581783.3612488

CCS CONCEPTS

• **Computing methodologies** → *Reconstruction.*

KEYWORDS

Scene text recognition, scene text image super-resolution, transformer.

ACM Reference Format:
Minyi Zhao, Shijie Xuyang, Jihong Guan, and Shuigeng Zhou. 2023. STIRER: A Unified Model for Low-Resolution Scene Text Image Recovery and Recognition. In *Proceedings of the 31st ACM International Conference on Multimedia (MM '23), October 29-November 3, 2023, Ottawa, ON, Canada.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3581783.3612488

1 INTRODUCTION

Scene text images contain rich semantic information that can be extracted and interpreted for various downstream applications like text-based understanding [30, 42, 55], and auto-driving [38]. Although scene text recognition (STR) from high-resolution (HR) images has achieved remarkable progress thanks to a series of advanced scene text recognizers [5, 7] developed via deep learning techniques, the performance of reading low-resolution (LR) scene text images is still undesirable, due to various image quality degradation factors such as blurry and low-contrast [45].

In recent years, there are increasing studies on the recognition of LR scene text images. As shown in Fig. 1, existing works roughly fall into two groups. The first group directly recognizes texts from LR images. Most early STR works can be classified into this group. However, as pointed out in some recent works [45, 56], due to the lack of sufficient pixel information, these methods are prone to misread the texts. Thereby, the texts will be incorrectly recognized (see Fig. 1(a)). The other group attempts to develop a scene text image super-resolution (STISR) [4, 6, 45] model to recover the missing details in LR images to improve the visual quality of the images, from which texts are subsequently recognized, thus boosting the recognition performance. More and more recently proposed models follow this paradigm. However, these STISR methods suffer from two shortcomings. First, some imperfect or even erroneous SR images may be mistakenly generated, which will mislead the subsequent recognition. For example, as shown in Fig. 1(b), the

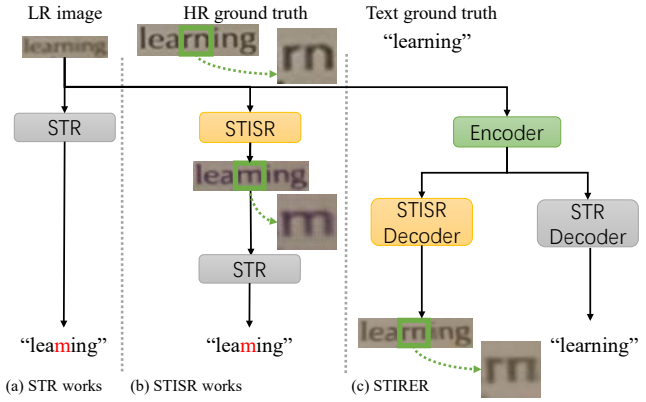

Figure 1: Architecture comparison of existing works (a) STR methods, (b) STISR methods, and (c) our model STIRER. STIRER employs a shared pixel feature encoder to extract text-specific features and two dedicated decoders to restore images and read texts respectively. The character strings at the bottom are recognition results, where red characters are incorrectly recognized, and black ones are correctly recognized. For the HR image, "rn" are zoomed.

characters "rn" are incorrectly recovered into 'm', which misleads the subsequent recognition. Second, in STISR works, as the STISR and STR models are jointly optimized, for pursuing high recognition accuracy, the fidelity of SR images may be impacted, which in turn degrades recognition. Again in Fig. 1(b), the contrast and the font color are enhanced to highlight the text, which obviously impairs the fidelity of the generated image. As a result, neither the recognition performance of STR nor the SR fidelity of STISR is desirable.

In this paper, we try to achieve both high recognition performance and super-resolution fidelity simultaneously on low-resolution text images. To this end, we propose a new model called **STIRER** (the abbreviation of **S**cene **T**ext **I**mage **RE**covery and **R**ecognition) to recover and recognize LR scene text images under a unified framework. As shown in Fig. 1(c), the basic idea of STIRER is first to obtain features that contain rich text-specific information, then two dedicated decoders are developed to simultaneously generate SR images and recognize the texts based on the encoded features and raw LR images. Here the two decoders are optimized separately to avoid malign competition, thus both recovery and recognition performance can be well optimized with the text-specific features from the encoder.

As high-quality text-specific features are essential to the recognition and recovery of LR images, accordingly, apart from implicitly learning the features from the feedback of the two decoders, we design a *progressive scene text swin transformer* (PSTST) architecture as the encoder to explicitly learn the features. Particularly, our PSTST module not only takes advantage of both self-attention and global contextual feature fusion to learn local and contextual pixel information, but also progressively learns text-specific information in each PSTST layer. Compared with existing STR and STISR methods, our STIRER model has two major advantages: 1) *high recognition accuracy and good fidelity*. We separately do STR and STISR and learn rich text-specific features to aid STR and STISR, which successfully avoids the zero-sum game between STR and STISR; 2) *high efficiency*. Thanks to our innovative design, our STR and STISR decoders can outperform the existing methods with much fewer parameters.

Contributions of this paper are as follows: 1) To address the drawbacks of existing STR and STISR methods, we propose a unified model STIRER to recover and recognize LR text images simultaneously. 2) We develop a progressive scene text swin transformer to serve as the encoder of STIRER to extract rich text-specific pixel features to aid both recognition and recovery. 3) We conduct extensive experiments on two LR datasets, which show the superiority of STIRER to the existing techniques on both recognition performance and super-resolution fidelity as well as efficiency.

2 RELATED WORK

2.1 Scene text image super-resolution

According to whether exploiting text-specific information from HR images, recent STISR methods can be roughly categorized into two types: generic super-resolution approaches and *scene text image super-resolution* (STISR) approaches. Generic image super-resolution methods usually use pixel loss functions (*e.g.* L_1 or L_2 loss) to capture pixel information for model supervision. For example, SRCNN [11, 34] designs various convolutional neural networks. [50] and SRResNet [22] adopt generative adversarial networks to generate distinguishable SR images. RCAN [53] and SAN [10] introduce attention mechanisms to promote image recovery. Recently, transformer-structured methods [23, 25, 49] are proposed to further boost performance.

Differently, STISR approaches focus on the extraction of various text-specific information to aid the model. Specifically, [14, 37, 46] calculate text-specific losses. Mou *et al.* [32] introduces a pluggable super-resolution module. Wang *et al.* [45] employs TSRN blocks and gradient profile loss to capture sequential information of text images and gradient fields of HR images for sharpening the texts, respectively. PCAN [54] proposes to learn sequence-dependent and high-frequency information for recovery. STT [4] uses a pre-trained transformer recognizer to conduct text-focused supervision. TG [6] and [33] exploit stroke-level information from HR images for more fine-grained super-resolution. TPGSR [28], TATT [29], C3-STISR [56], MARCONet [24] and DPMN [57] use various text-specific clues to guide the super-resolution. However, these methods face the risk of forcing the STR models to read erroneous SR images. In addition, they are also unable to balance recognition performance on LR images and fidelity of generated SR images well. Thus, neither text recognition ability nor HR fidelity is desirable.

2.2 Scene text recognition

Text recognition on HR images has made great progress in recent years [2, 5, 7–9]. Specifically, CRNN [40] takes CNN and RNN as the encoder and proposes a CTC-based [16] decoder. ASTER [41] and MORAN [27] propose to rectify texts. AutoSTR [52] employs neural architecture search (NAS) [13] to search backbones for text recognition. More recently, attention-based [18, 39, 51], semantic-based [36, 51], transformer-based [1], linguistical-based [15, 47],

and high-efficiency [3, 12] approaches are proposed to further lift the performance. Although these methods are able to handle irregular, occluded, and incomplete text images, they still have difficulty in recognizing low-resolution images due to the lack of sufficient pixel information. What is more, finetuning these recognizers is insufficient to accurately recognize texts from LR images, as reported in [45]. Therefore, existing STR techniques cannot handle text reading of LR images well.

2.3 Differences between existing works and our model

The differences lie in three aspects. Firstly, our aim is to recover and recognize low-resolution scene text images simultaneously and collaboratively under a unified framework , while STR is only for recognition, and STISR is mainly dedicated to recovery where STR is used to indirectly evaluate its performance. Secondly, different from recent STR methods, STIRER additionally provide essential text-specific information to aid text recognition. Thirdly, compared with STISR models that directly recognize texts from SR images that may carry noisy or erroneous information, STIRER develops an STR decoder to read texts from LR images with the help of text-specific feature. Moreover, we introduce a STISR decoder so that STIRER can concentrate on boosting the fidelity of SR images. As a result, STIRER outperforms STISR models on both recognition and fidelity. We also notice that some recent works have been proposed to recognize and recover LR images. In particular, PlugNet [32] employs a shared CNN backbone to extract features. Then, STR and STISR are performed on the features. However, these extracted features may also carry erroneous information. IFR [20] is proposed to iteratively recognize and recover the previous SR images. Nevertheless, IFR only extracts text-specific information to aid the super-resolution module and ignores the recognition module. Experimental results show that our method can achieve better performance.

3 MODEL

3.1 Overview

Given a low-resolution image $I_{LR} \in \mathbb{R}^{C\times N}$, where C is the number of channels of each image, $N = H \times W$ is the collapsed spatial dimension, H and W are the height and width of the LR image. The goal of STIRER is to produce a super-resolution (SR) image $I_{SR} \in \mathbb{R}^{C\times(4\times N)}$ based on the input LR image I_{LR} and read the texts on the LR image I_{LR}.

Fig. 2 shows the architecture of our model STIRER, which is composed of three major components: the *progress scene text swin transformer* (PSTST in short) ϕ (boxed in purple) and two decoders, namely the *STISR decoder* Φ_{SR} and the *STR decoder* Φ_{Rec}. Specifically, the LR image I_{LR} is first fed into the PSTST ϕ to encode a text-specific feature h_L (i.e., $h_L = \phi(I_{LR})$). Then, this feature is concatenated with the raw LR image I_{LR} to generate the corresponding SR image I_{SR}^{L+1} and predicted text distribution p_{L+1}, respectively. It is worth mentioning that our PSTST encoder ϕ consists of L basic PSTST blocks. As shown in Fig. 2, each block generates two intermediate results: an SR image and a predicted distribution, to enrich the representation of the clue. Thereby, a total of $L+1$ SR images, denoted as $\{I_{SR}^1, ..., I_{SR}^{L+1}\}$, and $L+1$ probability distributions, *i.e.,* $\{p_1, ..., p_{L+1}\}$, are generated. With these results, plus the HR image I_{HR} of each training LR image and its text-level annotation y, the super-resolution loss $\mathcal{L}_{SR}$ and the recognition loss $\mathcal{L}_{Rec}$ is evaluated for model training.

3.2 Progress scene text swin transformer

We start by giving a brief introduction to our PSTST used to encode a text-specific feature to aid the final decoding. The major motivation of our encoder is two-fold: (i) comprehensively learning both local and global contextual information from LR images and (ii) obtaining a rich text-specific feature for the subsequent recognition and recovery. For (i), both local and global contextual information are helpful for the model to estimate the blurry detail. However, it is not effective or efficient for existing methods to capture them. In particular, TBSRN [4] employs multi-head self-attention mechanism [43] to learn the dependencies of all the pixels but runs inefficiently. SwinIR [25] notices the inefficiency of typical self-attention and proposes to compute dependencies within a small window. Nevertheless, shifted window-based attention cannot grasp global contextual information well, and stacking a number of shifted window attention layers is also inefficient. As for (ii), as pointed out in [29, 56], a highly representative feature that carries rich text-specific information is beneficial to recovery. Therefore, to capture both local and global dependencies and enrich the encoded features in an economic manner, we design a *progressive scene text swin transformer*. As shown in Fig. 2, our PSTST consists of a convolutional layer and L basic PSTST blocks. The first convolutional layer is used to generate an initial feature $h_0 \in \mathbb{R}^{C'\times H\times W}$, where C' is the channel number of the feature. Then, L basic PSTST blocks are stacked to learn highly representative features. Each basic block takes the former feature as input and generates the next feature and two intermediate results. Taking the i-th block ϕ_i for instance, we have:

$$h_i, I_{SR}^i, p_i = \phi_i(h_{i-1}), \tag{1}$$

where I_{SR}^i and p_i are the resulting intermediate SR image and predicted distribution, respectively. In particular, as shown in Fig. 2, we first use N_i swin transformer layers [25] (STL) to conduct shifted window attention with a window size of $[H, \frac{W}{32}]$, which helps the model to grasp adjacent pixel information. Then, the preliminary feature is passed to a Bi-GRU network to learn contextual information. To better enrich the representation of the feature h_i, we introduce two mini decoders to each PSTST basic block for the generation of I_{SR}^i and p_i. Let the STISR decoder and the STR decoder in the i-th basic block be ϕ_i^{SR} and ϕ_i^{Rec}, respectively, we generate the intermediate results as follows:

$$I_{SR}^i = \phi_i^{SR}(h_i), \tag{2}$$

$$p_i = \phi_i^{Rec}(h_i). \tag{3}$$

We use several simple convolutional layers and six transformer encoder layers with a full connection layer to implement the STISR decoder ϕ_i^{SR} and the STR decoder ϕ_i^{Rec}. Then, we progressively put supervision on these intermediate results to generate the representation of the feature h_i. The details of the loss functions are given in Sec. 3.5.

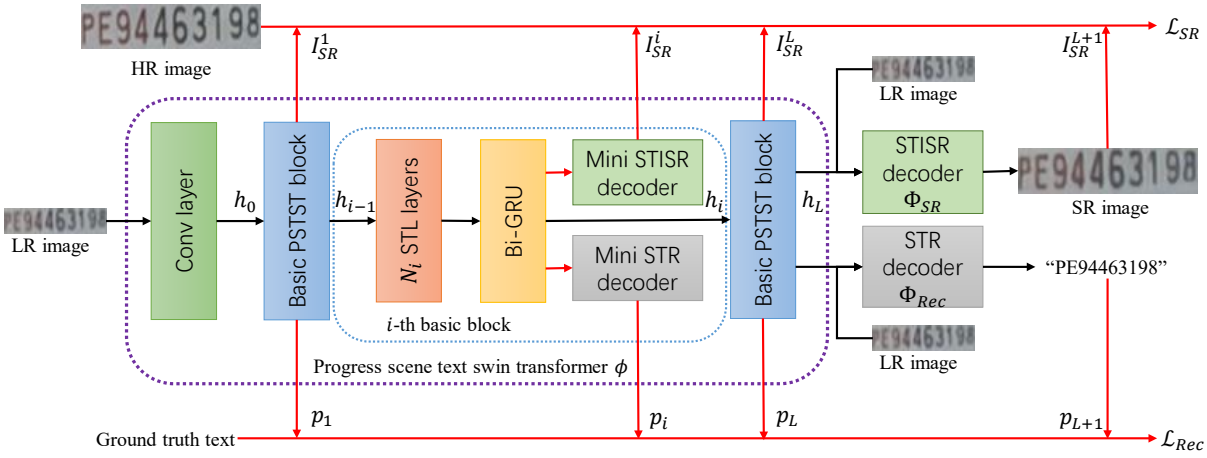

Figure 2: The architecture of our model STIRER. Red lines are only valid during training.

3.3 STISR decoder

The aim of the STISR decoder Φ_{SR} in STIRER is to generate a high fidelity SR image I_{SR}^{L+1} according to the raw LR image and the encoded feature h_L, which can be written as follows:

$$I_{SR}^{L+1} = \Phi_{SR}(I_{LR}, h_L). \tag{4}$$

In our implementation, we concatenate the LR image and the encoded feature at the channel dimension and feed them into three SRB [45] blocks to obtain the final SR image.

3.4 STR decoder

In this section, we introduce the design of the STR decoder Φ_{Rec} used in STIRER. The goal of the STR decoder is to read the text p_L from the LR image I_{LR} with the help of the feature h_L:

$$p_L = \Phi_{Rec}(I_{LR}, h_L). \tag{5}$$

Note that pixel information is valuable for the recognition of LR images. Therefore, utilizing down-sampling layers as in typical STR works [12, 40] is inappropriate for LR images. Thus, we first crop the input image and the encoded feature to various patches in the shape of $[H, \frac{W}{32}]$, then use a transformer decoder containing six encoder layers and six decoder layers to auto-regressively generate the text distribution:

$$p_L(y|[I_{LR}, h_L]) = \prod_{t=1}^{T} p_L(y_t|y_{<t}, [I_{LR}, h_L]), \tag{6}$$

where y is the ground truth text and T is the number of decoding steps of the STR decoder. Such an STR decoder can not only take full use of pixel information with the self-attention mechanism, but also accurately decode texts with the help of previous predictions.

3.5 Loss functions

We evaluate two losses in STIRER to supervise model training. The first one is the super-resolution loss $\mathcal{L}_{SR}$, which can be calculated via the gradient profile loss $\mathcal{L}_{GP}$ [45] between the HR image I_{HR} and all the SR images:

$$\mathcal{L}_{SR} = \Sigma_i i \mathcal{L}_{GP}(I_{SR}^i, I_{HR}). \tag{7}$$

Algorithm 1 The training procedure of STIRER.

1: **Input:** $\phi, \Phi_{SR}, \Phi_{Rec}, I_{LR}, I_{HR}, y, L$
2: Generate initial h_0 according to Sec. 3.2
3: **for** i in $[1, ..., L]$ **do**
4: $\quad h_i = \phi_i(h_{i-1})$
5: $\quad I_{SR}^i = \phi_i^{SR}(h_i)$
6: $\quad p_i = \phi_i^{Rec}(h_i)$
7: $I_{SR}^{L+1} = \Phi_{SR}(I_{LR}, h_L)$
8: $p_L = \Phi_{Rec}(I_{LR}, h_L)$
9: Compute the loss $\mathcal{L}$ via Eq. (9)
10: **return** $\mathcal{L}$

A progressive weight i is used to make the model put more emphasis on deeper layers and the last decoded SR image I_{SR}^{L+1}. Furthermore, a recognition loss $\mathcal{L}_{Rec}$ is calculated to supervise the STR decoder. Two recognition losses are utilized for different purposes. One is the CTC loss $\mathcal{L}_{CTC}$ used to supervise the pixel features $\{h_1, ..., h_L\}$. Another is the cross-entropy loss $\mathcal{L}_{CE}$ for the STR decoder Φ_{Rec}. The overall recognition loss can be written as follows:

$$\mathcal{L}_{Rec} = \Sigma_i i \mathcal{L}_{CTC}(p_i, y) + (L+1)\mathcal{L}_{CE}(p_{L+1}, y). \tag{8}$$

Above, we also introduce the progressive factor i to the recognition loss. Thereby, we progressively enhance the representation of the feature h_i with the supervision signal from the super-resolution loss and recognition loss.

The final loss of STIRER is the combination of the super-resolution loss $\mathcal{L}_{SR}$ and the recognition loss $\mathcal{L}_{Rec}$:

$$\mathcal{L} = \mathcal{L}_{SR} + \mathcal{L}_{Rec}. \tag{9}$$

3.6 Overall training and evaluation procedures

Here, we describe the overall training procedure of STIRER to better describe the implementation. As presented in Alg. 1, we first encode the pixel feature by the PSTST ϕ (Line 2~Line 6). In particular, we first generate the initial feature h_0 via a simple convolutional layer

(Line 2). Then, we use L basic PSTST blocks to progressively produce the pixel feature h_i (Line 4). We also generate two intermediate results (Line 5∼Line 6) for better representing h_i. After generating the final feature h_t, two decoders are applied to recovering and reading the LR image I_{LR} (Line 7∼Line 8). Last, we collect all the SR images (I_{SR}^1, ..., I_{SR}^{L+1}) and the predicted distributions (p_1, ..., p_{L+1}) to compute the loss for model training (Line 9).

As for performance evaluation, we need just to ignore the intermediate results (Line 5∼Line 6 in Alg. 1).

4 PERFORMANCE EVALUATION

4.1 Datasets and metrics

As in recent STR and STISR works [4, 29, 40, 56], we use the synthetic datasets to pre-train our model and then finetune the model on the low-resolution datasets. The synthetic datasets include Synth90K [19] and SynthText [17]. Two real-world low-resolution datasets are used: TextZoom and IC15-352. Besides, following [4, 6], we also manually generate LR images from SVT [44], IIIT [31], IC15 [21], and SVTP [35] for generalization ability evaluation. In what follows, we give a brief review of the low-resolution datasets used for finetuning and performance evaluation.

Five metrics are used in our paper to evaluate the performance of the model from the aspects of recognition ability, fidelity, and efficiency, accordingly. The first metric is *word-level recognition accuracy* which evaluates the recognition performance of various methods. Following the settings of previous works [6, 28, 29, 56], we remove punctuation and convert uppercase letters to lowercase letters for calculating recognition accuracy. Besides, we report *Peak Signal-to-Noise Ratio* (PSNR) and *Structure Similarity Index Measure* (SSIM) [48] as the metrics to measure fidelity. As for efficiency, we calculate the *Floating-point Operations per Second* (FLOPs) and the number of parameters (Params) of each model.

4.2 Implementation details

All experiments are conducted on 4 NVIDIA RTX 3090 GPUs with 24GB memory. The PyTorch version is 1.10. STIRER is trained by using the AdamW [26] optimizer. We set different learning rates for different STIRER modules. During pre-training, the learning rate for the STISR decoder Φ_{SR} is 5×10^{-4} while 10^{-4} for the rest. When it comes to finetuning, the learning rate for the encoder and the STR decoder is reduced to 5×10^{-5}. The batch size is set to 256 for pre-training and 64 for finetuning. The intermediate channel number C' is set to 48 for efficiency. L and N_i are set to 3 and [4, 5, 6], respectively. Following [6, 12, 40, 56], we use a charset (*i.e.,* letters and numbers) of length 36.

4.3 Comparing with SOTA approaches

4.3.1 Results on TextZoom. Here, we evaluate our STIRER model on the **TextZoom** dataset and compare the model performance with six STR approaches and three STISR methods. Specifically, the recognizers including CRNN [40], MORAN [27], SEED [36], ASTER [41], SVTR [12] and ABINet [15] while three recent STISR methods are TG [6], TPGSR [28], and C3-STISR [56]. We also compare our method with IFR [20]. All the experimental results are presented in Tab. 1.

We first check the performance of existing STR methods. From the 2nd row to the 10th row of Tab. 1, we can see that typical STR methods cannot well recognize low-resolution scene text images. For example, CRNN, the state-of-the-art recognizer SVTR and ABINet can only correctly read 27.3%, 50.8% and 60.0% images, respectively. What makes the matter worse is that finetuning these recognizers on low-resolution datasets also cannot address the recognition issue. For instance, after finetuning, the recognition performance of CRNN, SVTR and ABINet is boosted from 27.3%, 41.1%, 50.8% to 61.7% and 60.0% to 69.0%, respectively, which are still unsatisfactory.

Then, we pay attention to typical STISR methods. Since STISR methods naturally cannot recognize scene texts, we use three recognizers, including two efficient recognizers CRNN and SVTR, and ABINet to work with STISR for recognition. From the 11th to 19th rows we can see that (1) These STISR methods succeed in boosting the recognition of LR images. For example, as can be seen from the 11th row to the 13th row, TG, TPGSR and C3-STISR lift the accuracy of CRNN from 27.3% to 48.9%, 51.8%, and 53.7%, respectively. (2) STISR approaches may generate imperfect results, which will mislead the subsequent recognition. For instance, as can be seen in the 9th and the 14th rows, the performance of TG+SVTR (61.6%) is inferior to that of directly finetuning SVTR, which has an accuracy of 61.7%.

Finally, we analyze our STIRER method. To better justify the model performance under a fair comparison, we additionally design a variant that utilizes CRNN to implement the STR decoder Φ_{Rec} of STIRER. The performance of this variant is given in the 21st row of Tab. 1, while from the 22nd row to the 23rd row, we present the performance of our standard STIRER method. We first check the recognition ability of STIRER. From Tab. 1 we can see that the best recognition performance is obtained by our STIRER method. In particular, compared with all the methods that utilize CRNN for recognition, our STIRER variant significantly lifts the average accuracy from 53.7% (see the 13th row) to 68.6%. Besides, our standard STIRER method has an average accuracy of 74.8%, which is 5.2% higher than the SOTA method IFR. Then, we pay attention to the fidelity performance. Apparently, the best fidelity performance is also achieved by our technique. In particular, all our various STIRER variants (from the 21st row to the 23rd row) promote the PSNR from 21.51 (19th row) to 22.32, 21.98 and 22.46, respectively. Last, we further explore the advantage of STIRER by taking efficiency into consideration. As can be seen from Tab. 1, STIRER achieves significant accuracy and fidelity improvement with much less computational cost. Concretely, the FLOPs of STIRER (21st row) is 0.74, which is about 15 times lower than the combination of C3-STISR and ABINet (17th row). To better highlight the model performance of various methods, we plot the accuracy-FLOPs diagram for all methods in Fig. 3. As shown in Fig. 3, our STIRER locates in the left-top corner, which demonstrates its highest accuracy and efficiency. All these results show the superiority of our STIRER method in recognition accuracy, fidelity performance, and efficiency.

4.3.2 Performance improvement on other STR datasets. We also evaluate the related approaches on more STR datasets, including IC15-352 [6], SVT [44], IIIT [31], IC15 [21], and SVTP [35] to compare their generalization abilities. Following [6, 28], we manually

Table 1: Performance comparison with SOTAs on the TextZoom dataset. "Recognizer" means the recognizer used by a STISR method. "Dataset" indicates the dataset used to develop the model. Concretely, 'S': synthetic dataset; 'L': low-resolution dataset; 'T': additional linguistical dataset used for pre-training.

ID	Method	Recognizer	Dataset	Recognition accuracy				Fidelity		Efficiency	
				Easy	Medium	Hard	Average	PSNR (dB)	SSIM	FLOPs (G)	Params (M)
1	LR	-	-	-	-	-	-	20.35	0.6961	-	-
2	CRNN	-	S	37.5%	21.4%	21.1%	27.3%	-	-	0.91	8.36
3	MORAN	-	S	56.2%	35.9%	28.2%	41.1%	-	-	0.93	20.40
4	SEED	-	S+T	63.5%	41.3%	29.3%	45.8%	-	-	5.30	24.99
5	ASTER	-	S	66.4%	43.4%	32.3%	48.5%	-	-	6.22	20.99
6	SVTR	-	S	69.1%	45.6%	34.1%	50.8%	-	-	1.90	22.70
7	ABINet	-	S+T	76.5%	57.0%	43.5%	60.0%	-	-	7.29	36.86
8	CRNN	-	S+L	58.4%	34.1%	27.6%	41.1%	-	-	0.91	8.36
9	SVTR	-	S+L	80.0%	59.0%	42.4%	61.7%	-	-	1.90	22.70
10	ABINet	-	S+T+L	82.3%	69.5%	52.4%	69.0%	-	-	7.29	36.86
11	TG	CRNN	S+L	61.2%	47.6%	35.5%	48.9%	21.40	0.7456	1.82	9.19
12	TPGSR	CRNN	S+L	63.1%	52.0%	38.6%	51.8%	21.18	**0.7762**	4.75	35.33
13	C3-STISR	CRNN	S+L	65.2%	53.6%	39.8%	53.7%	21.51	0.7721	4.54	58.52
14	TG	SVTR	S+L	76.8%	61.5%	43.3%	61.6%	21.40	0.7456	2.81	23.53
15	TPGSR	SVTR	S+L	80.9%	62.8%	46.4%	64.5%	21.18	**0.7762**	5.74	49.67
16	C3-STISR	SVTR	S+L	81.6%	63.8%	46.7%	65.1%	21.51	0.7721	5.53	72.86
17	TG	ABINet	S+T+L	78.6%	67.7%	49.2%	66.0%	21.40	0.7456	8.20	37.69
18	TPGSR	ABINet	S+T+L	83.5%	67.9%	52.2%	67.4%	21.18	**0.7762**	11.13	99.83
19	C3-STISR	ABINet	S+T+L	80.1%	68.5%	51.9%	67.7%	21.51	0.7721	10.92	123.02
20	IFR	-	S+L	83.0%	69.7%	53.2%	69.6%	21.23	0.7407	-	-
21	STIRER	CRNN	S+L	82.1%	68.9%	52.1%	68.6%	22.32	0.7597	1.67	9.51
22	STIRER	-	S	77.2%	60.0%	44.7%	61.7%	21.98	0.7332	0.74	4.03
23	STIRER	-	S+L	**87.9%**	**74.2%**	**59.6%**	**74.8%**	**22.46**	0.7706	**0.74**	**4.03**

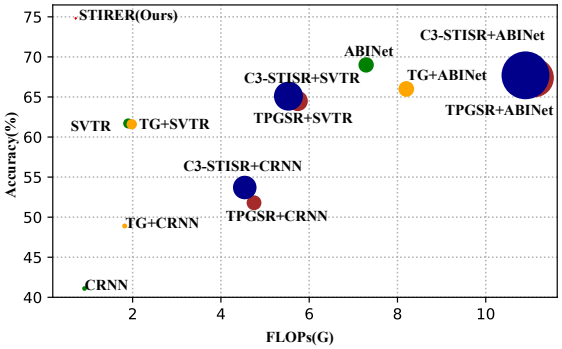

Figure 3: The accuracy-efficiency diagram for all methods on the TextZoom dataset. The diameter of each circle represents the parameter size of the corresponding method. STIRER is at the best place.

generate LR images for SVT, IIIT, IC15, and SVTP by 4× downscaling. Here we report only accuracy to compare the recognition performance since these datasets do not contain LR-HR image pairs. All the experimental results are presented in Tab. 2. From Tab. 2 we can see that the SOTA STISR method C3-STISR generalizes poorly on these datasets. And obviously, our method outperforms SVTR,

Table 2: Performance comparison with various state-of-the-art methods on the IC15-352, SVT, IIIT, IC15, and SVTP datasets.

Method	IC15-352	SVT	IIIT	IC15	SVTP
SVTR	66.5%	49.8%	56.2%	64.8%	39.4%
+C3-STISR	75.0%	47.4%	50.0%	61.1%	37.5%
ABINet	80.0%	66.1%	53.4%	69.3%	52.5%
+C3-STISR	76.6%	64.4%	61.7%	69.7%	49.1%
STIRER	**83.0%**	**67.4%**	**65.8%**	**72.3%**	**55.0%**

C3-STISR and ABINet on all five datasets, which justifies the strong generalization ability of STIRER.

4.4 Visualization

We visualize some results of our STIRER model and compare them with that of existing STISR methods to better demonstrate the advantage of our model. 8 typical cases are illustrated in Fig. 4. The LR and HR columns in Fig. 4 denote the input LR and HR images, and the texts below them are recognized by SVTR. The texts in the STIRER column are recognized by our STR decoder. From Fig. 4 we can see that (1) Directly reading texts from LR images is not an easy task due to the lack of pixel information. For example, in the 1st and 2nd cases, texts are incorrectly recognized. (2) Although applying

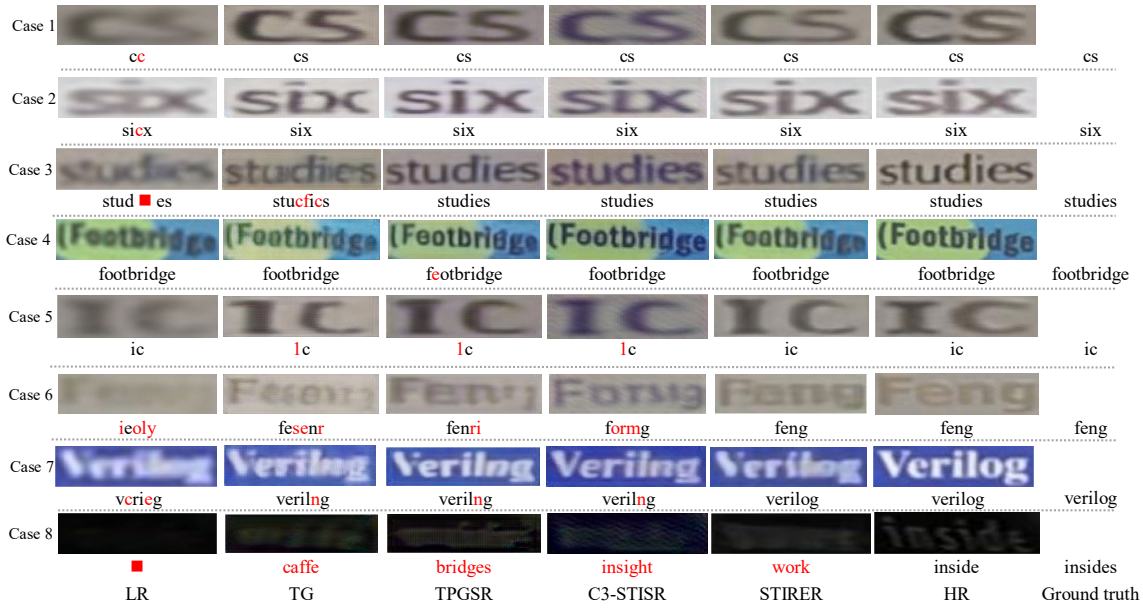

Figure 4: Visual examples of generated images and recognition results of our STIRER and some existing STISR methods. Red/black characters are incorrectly/correctly recognized results. Texts below images in the LR, TG, TPGSR, C3-STISR and HR columns are the corresponding recognition results obtained by SVTR.

STISR methods to generate SR images can alleviate the recognition challenge to some degree, STISR methods have two major shortcomings. On the one hand, STISR methods may generate imperfect or even wrong results. For example, as shown in the 3rd, 4th and 5th cases, STISR methods mistakenly recover some characters, which leads to the failure of the subsequent recognition. On the other hand, STISR methods also may impair the fidelity of the SR images. For example, as shown in the 5th case, both TPGSR and C3-STISR change the font color and the contrast of the image. Although they can better highlight the text in the image, a different style from the HR image will impair fidelity. These cases help to explain the reason why typical STISR methods cannot balance recognition accuracy and fidelity well. Ergo, in this paper we propose STIRER to do STISR and STR simultaneously and collaboratively, but separately. As shown in Fig. 4, STIRER succeeds in avoiding the malign competition between recovery and recognition. Specifically, the SR images generated by STIRER maintain consistency of text style such as foreground, background, font color etc. What is more, correct recognition results are also simultaneously obtained. This helps to justify the outstanding performance of the STIRER model. (3) The failure case demonstrates the limitation of our method. As can be seen in the 8th case, in some tough situations where pixel information is extremely insufficient, the recovery and the recognition may fail. This indicates that the recovery and recognition of LR images is still a challenging task.

4.5 Ablation study

Here we conduct ablation study to check the effectiveness of our model design. We use the **TextZoom** dataset and present all the experimental results in Tab. 3.

Table 3: Ablation study on the progress scene text swin transformer, STISR decoder and STR decoder.

Method	Accuracy	PSNR(dB)
STIRER	74.8%	22.46
Without PSTST	67.5%	21.76
Without mini decoders in PSTST	74.1%	22.34
Without STL layers in PSTST	72.7%	22.13
Without Bi-GRUs in PSTST	70.6%	22.00
Without STISR decoder	73.1%	-
Without STR decoder	-	22.14

Table 4: Performance of the intermediate results generated by PSTST.

i	Recognition accuracy of p_i	PSNR of I^i_{SR}
1	65.3%	20.84
2	68.3%	20.77
3	67.6%	21.20
4	74.8%	22.46

4.5.1 Effect of progressive scene text swin transformer. We start by checking the effect of our PSTST used to extract a text-specific feature for decoding. The core idea of PSTST is two-fold: a progressive mechanism used to provide direct supervision for each PSTST block and the combination of STL and Bi-GRU blocks to capture both local and contextual information. To this end, four PSTST variants are considered to comprehensively check PSTST performance. In the first variant, we remove the entire PSTST, which means we directly use the STISR decoder and the STR decoder to generate the final results. As shown in the 3rd row of Tab. 3, the accuracy and the PSNR significantly deteriorate from 74.8% and 22.46 to 67.5%

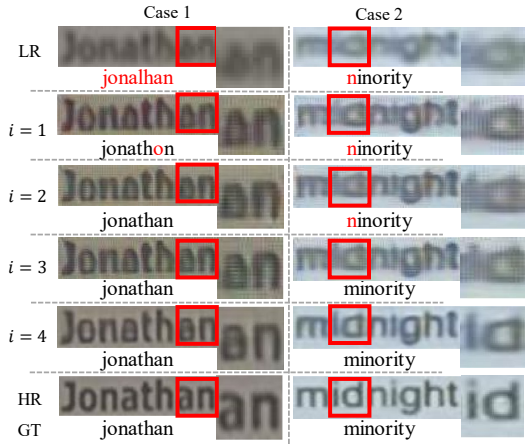

Figure 5: Examples of the intermediate recognition results and SR images generated by STIRER. Here, GT denotes ground truth while red/black characters are incorrectly/correctly recognized ones.

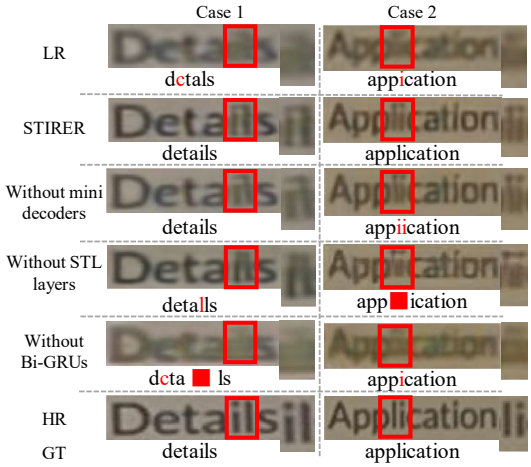

Figure 6: Visual examples of various STIRER variants. GT denotes ground truth while red/black characters are incorrectly/correctly recognized ones.

and 21.76. This indicates that PSTST can provide useful features to aid the decoding. Then, we design a variant that removes the mini decoders in PSTST, which means we do not compute losses to directly supervise the features extracted in each PSTST block. As shown in the 4th row of Tab. 3, obviously, the performance is also degraded. Last, we check the performance of the STL layers and the Bi-GRUs used in PSTST. As presented in the 5th and 6th rows of Tab. 3, the model performance is impaired. These results demonstrate the design of our STIRER method.

Furthermore, we report the performance of intermediate results progressively extracted by each PSTST block. As shown in Tab. 4, the recognition accuracy and PSNR of these intermediate results are basically gradually improved. This validates that PSTST has the ability to encode rich text-specific features for decoding through each block in a progressive manner. We also conduct a visualization study to check these intermediate results. As shown in Fig. 5, the quality of intermediate results is gradually refined. For example, in the 1st case, the characters "an" are successfully recovered in the 2nd PSTST block while the character 'm' in the 2nd case is recovered in the 3rd block.

Table 5: Computational cost and parameter size of each STIRER module.

Metrics	Feature encoder ϕ	STISR decoder Φ_{SR}	STR decoder Φ_{Rec}
FLOPs (G)	0.52	0.13	0.09
Params (M)	0.52	0.13	3.38

4.5.2 Effects of the STISR decoder and STR decoder. In STIRER, we utilize two decoders to do two tasks. To check whether one decoder will influence the other, we design two variants that only use one decoder to do one task. As presented in the 7th and 8th rows of Tab. 3, the accuracy/PSNR performance of these two variants is inferior to that of the whole STIRER model: 73.1%/22.14 v.s. 74.8%/22.46. This indicates that (1) Learning the recognition task can provide useful information for the recovery tasks and vice versa; (2) Doing STISR and STR collaboratively in a unified framework is better than doing STISR and STR with two independent models.

4.5.3 Visualization study. We also visualize some SR images generated by the aforementioned variants in Fig. 6. We can see that the recovery performance of these variants is inferior to that of the whole STIRER model. In the 2nd case, these variants mistakenly recover the characters "li" into "ii", 'i' and 'i', respectively.

4.5.4 Efficiency of each STIRER module. Here we report the efficiency results of each STIRER module in Tab. 5. STIRER costs only 0.52G FLOPs to encode the text-specific features, and uses 0.13G and 0.09G FLOPs to decode the final results.

5 CONCLUSION

In this paper, a new model called STIRER is presented to do low-resolution scene text recovery and recognition simultaneously and collaboratively under a unified framework. Concretely, STIRER first uses a progressive scene text swin transformer to extract text-specific features. Then, two decoders are designed to generate SR images and recognition results based on the encoded features and the raw LR images for the corresponding STISR and STR tasks. In such a unified manner, STIER not only successfully avoids malign competition between STISR and STR but also boosts the performance of STISR and STR by providing essential information. Extensive experiments on two low-resolution scene text datasets and several manually generated low-resolution datasets validate the superiority of our method to the existing approaches in recognition accuracy and super-resolution fidelity as well as efficiency.

ACKNOWLEDGMENTS

Jihong Guan was supported by National Natural Science Foudnation of China (NSFC) under grant No. U1936205.

REFERENCES

[1] Rowel Atienza. 2021. Vision transformer for fast and efficient scene text recognition. In *International Conference on Document Analysis and Recognition*. Springer, 319–334.

[2] Fan Bai, Zhanzhan Cheng, Yi Niu, Shiliang Pu, and Shuigeng Zhou. 2018. Edit probability for scene text recognition. In *CVPR*. 1508–1516.

[3] Darwin Bautista and Rowel Atienza. 2022. Scene Text Recognition with Permuted Autoregressive Sequence Models. In *Proceedings of the 17th European Conference on Computer Vision (ECCV)*. Springer International Publishing, Cham.

[4] Jingye Chen, Bin Li, and Xiangyang Xue. 2021. Scene Text Telescope: Text-Focused Scene Image Super-Resolution. In *CVPR*. 12026–12035.

[5] Jingye Chen, Haiyang Yu, Jianqi Ma, Mengnan Guan, Xixi Xu, Xiaocong Wang, Shaobo Qu, Bin Li, and Xiangyang Xue. 2021. Benchmarking Chinese Text Recognition: Datasets, Baselines, and an Empirical Study. *arXiv preprint arXiv:2112.15093* (2021).

[6] Jingye Chen, Haiyang Yu, Jianqi Ma, Bin Li, and Xiangyang Xue. 2022. Text gestalt: Stroke-aware scene text image super-resolution. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 285–293.

[7] Xiaoxue Chen, Lianwen Jin, Yuanzhi Zhu, Canjie Luo, and Tianwei Wang. 2021. Text recognition in the wild: A survey. *ACM Computing Surveys (CSUR)* 54, 2 (2021), 1–35.

[8] Zhanzhan Cheng, Fan Bai, Yunlu Xu, Gang Zheng, Shiliang Pu, and Shuigeng Zhou. 2017. Focusing attention: Towards accurate text recognition in natural images. In *ICCV*. 5076–5084.

[9] Zhanzhan Cheng, Yangliu Xu, Fan Bai, Yi Niu, Shiliang Pu, and Shuigeng Zhou. 2018. Aon: Towards arbitrarily-oriented text recognition. In *CVPR*. 5571–5579.

[10] Tao Dai, Jianrui Cai, Yongbing Zhang, Shu-Tao Xia, and Lei Zhang. 2019. Second-order attention network for single image super-resolution. In *CVPR*. 11065–11074.

[11] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. 2015. Image super-resolution using deep convolutional networks. *TPAMI* 38, 2 (2015), 295–307.

[12] Yongkun Du, Zhineng Chen, Caiyan Jia, Xiaoting Yin, Tianlun Zheng, Chenxia Li, Yuning Du, and Yu-Gang Jiang. 2022. SVTR: Scene Text Recognition with a Single Visual Model. *arXiv preprint arXiv:2205.00159* (2022).

[13] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. 2019. Neural architecture search: A survey. *JMLR* 20, 1 (2019), 1997–2017.

[14] Chuantao Fang, Yu Zhu, Lei Liao, and Xiaofeng Ling. 2021. TSRGAN: Real-world text image super-resolution based on adversarial learning and triplet attention. *Neurocomputing* 455 (2021), 88–96.

[15] Shancheng Fang, Hongtao Xie, Yuxin Wang, Zhendong Mao, and Yongdong Zhang. 2021. Read Like Humans: Autonomous, Bidirectional and Iterative Language Modeling for Scene Text Recognition. In *CVPR*. 7098–7107.

[16] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *ICML*. 369–376.

[17] Ankush Gupta, Andrea Vedaldi, and Andrew Zisserman. 2016. Synthetic data for text localisation in natural images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2315–2324.

[18] Wenyang Hu, Xiaocong Cai, Jun Hou, Shuai Yi, and Zhiping Lin. 2020. Gtc: Guided training of ctc towards efficient and accurate scene text recognition. In *AAAI*, Vol. 34. 11005–11012.

[19] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Synthetic data and artificial neural networks for natural scene text recognition. *arXiv preprint arXiv:1406.2227* (2014).

[20] Zhiwei Jia, Shugong Xu, Shiyi Mu, Yue Tao, Shan Cao, and Zhiyong Chen. 2021. IFR: Iterative Fusion Based Recognizer for Low Quality Scene Text Recognition. In *Pattern Recognition and Computer Vision: 4th Chinese Conference, PRCV 2021, Beijing, China, October 29–November 1, 2021, Proceedings, Part II 4*. Springer, 180–191.

[21] Dimosthenis Karatzas, Lluis Gomez-Bigorda, Anguelos Nicolaou, Suman Ghosh, Andrew Bagdanov, Masakazu Iwamura, Jiri Matas, Lukas Neumann, Vijay Ramaseshan Chandrasekhar, Shijian Lu, et al. 2015. ICDAR 2015 competition on robust reading. In *ICDAR*. IEEE, 1156–1160.

[22] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. 2017. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*. 4681–4690.

[23] Wenbo Li, Xin Lu, Jiangbo Lu, Xiangyu Zhang, and Jiaya Jia. 2021. On efficient transformer and image pre-training for low-level vision. *arXiv preprint arXiv:2112.10175* (2021).

[24] Xiaoming Li, Wangmeng Zuo, and Chen Change Loy. 2023. Learning Generative Structure Prior for Blind Text Image Super-resolution. *arXiv preprint arXiv:2303.14726* (2023).

[25] Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. 2021. Swinir: Image restoration using swin transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 1833–1844.

[26] Ilya Loshchilov and Frank Hutter. 2018. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*.

[27] Canjie Luo, Lianwen Jin, and Zenghui Sun. 2019. Moran: A multi-object rectified attention network for scene text recognition. *PR* 90 (2019), 109–118.

[28] Jianqi Ma, Shi Guo, and Lei Zhang. 2023. Text prior guided scene text image super-resolution. *IEEE Transactions on Image Processing* 32 (2023), 1341–1353.

[29] Jianqi Ma, Zhetong Liang, and Lei Zhang. 2022. A Text Attention Network for Spatial Deformation Robust Scene Text Image Super-resolution. In *CVPR*. 5911–5920.

[30] Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. 2021. Docvqa: A dataset for vqa on document images. In *CVPR*. 2200–2209.

[31] Anand Mishra, Karteek Alahari, and CV Jawahar. 2012. Top-down and bottom-up cues for scene text recognition. In *CVPR*. IEEE, 2687–2694.

[32] Yongqiang Mou, Lei Tan, Hui Yang, Jingying Chen, Leyuan Liu, Rui Yan, and Yaohong Huang. 2020. Plugnet: Degradation aware scene text recognition supervised by a pluggable super-resolution unit. In *ECCV*. Springer, 158–174.

[33] Shimon Nakaune, Satoshi Iizuka, and Kazuhiro Fukui. 2021. Skeleton-aware Text Image Super-Resolution. (2021).

[34] Ram Krishna Pandey, K Vignesh, AG Ramakrishnan, et al. 2018. Binary document image super resolution for improved readability and OCR performance. *arXiv preprint arXiv:1812.02475* (2018).

[35] Trung Quy Phan, Palaiahnakote Shivakumara, Shangxuan Tian, and Chew Lim Tan. 2013. Recognizing text with perspective distortion in natural scenes. In *ICCV*. 569–576.

[36] Zhi Qiao, Yu Zhou, Dongbao Yang, Yucan Zhou, and Weiping Wang. 2020. Seed: Semantics enhanced encoder-decoder framework for scene text recognition. In *CVPR*. 13528–13537.

[37] Rui Qin, Bin Wang, and Yu-Wing Tai. 2022. Scene Text Image Super-Resolution via Content Perceptual Loss and Criss-Cross Transformer Blocks. *arXiv preprint arXiv:2210.06924* (2022).

[38] Sangeeth Reddy, Minesh Mathew, Lluis Gomez, Marçal Rusinol, Dimosthenis Karatzas, and CV Jawahar. 2020. Roadtext-1k: Text detection & recognition dataset for driving videos. In *ICRA*. IEEE, 11074–11080.

[39] Fenfen Sheng, Zhineng Chen, and Bo Xu. 2019. NRTR: A no-recurrence sequence-to-sequence model for scene text recognition. In *ICDAR*. IEEE, 781–786.

[40] Baoguang Shi, Xiang Bai, and Cong Yao. 2016. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *TPAMI* 39, 11 (2016), 2298–2304.

[41] Baoguang Shi, Mingkun Yang, Xinggang Wang, Pengyuan Lyu, Cong Yao, and Xiang Bai. 2018. Aster: An attentional scene text recognizer with flexible rectification. *TPAMI* 41, 9 (2018), 2035–2048.

[42] Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. 2019. Towards vqa models that can read. In *CVPR*. 8317–8326.

[43] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[44] Kai Wang, Boris Babenko, and Serge Belongie. 2011. End-to-end scene text recognition. In *ICCV*. IEEE, 1457–1464.

[45] Wenjia Wang, Enze Xie, Xuebo Liu, Wenhai Wang, Ding Liang, Chunhua Shen, and Xiang Bai. 2020. Scene text image super-resolution in the wild. In *ECCV*. Springer, 650–666.

[46] Wenjia Wang, Enze Xie, Peize Sun, Wenhai Wang, Lixun Tian, Chunhua Shen, and Ping Luo. 2019. Textsr: Content-aware text super-resolution guided by recognition. *arXiv preprint arXiv:1909.07113* (2019).

[47] Yuxin Wang, Hongtao Xie, Shancheng Fang, Jing Wang, Shenggao Zhu, and Yongdong Zhang. 2021. From two to one: A new scene text recognizer with visual language modeling network. In *ICCV*. 14194–14203.

[48] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. 2004. Image quality assessment: from error visibility to structural similarity. *TIP* 13, 4 (2004), 600–612.

[49] Zhendong Wang, Xiaodong Cun, Jianmin Bao, Wengang Zhou, Jianzhuang Liu, and Houqiang Li. 2022. Uformer: A general u-shaped transformer for image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 17683–17693.

[50] Xiangyu Xu, Deqing Sun, Jinshan Pan, Yujin Zhang, Hanspeter Pfister, and Ming-Hsuan Yang. 2017. Learning to super-resolve blurry face and text images. In *ICCV*. 251–260.

[51] Deli Yu, Xuan Li, Chengquan Zhang, Tao Liu, Junyu Han, Jingtuo Liu, and Errui Ding. 2020. Towards accurate scene text recognition with semantic reasoning networks. In *CVPR*. 12113–12122.

[52] Hui Zhang, Quanming Yao, Mingkun Yang, Yongchao Xu, and Xiang Bai. 2020. AutoSTR: Efficient Backbone Search for Scene Text Recognition. In *ECCV*.

[53] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. 2018. Image super-resolution using very deep residual channel attention networks. In *ECCV*. 286–301.

[54] Cairong Zhao, Shuyang Feng, Brian Nlong Zhao, Zhijun Ding, Jun Wu, Fumin Shen, and Heng Tao Shen. 2021. Scene Text Image Super-Resolution via Parallelly Contextual Attention Network. In *MM*. 2908–2917.

[55] Minyi Zhao, Bingjia Li, Jie Wang, Wanqing Li, Wenjing Zhou, Lan Zhang, Shijie Xuyang, Zhihang Yu, Xinkun Yu, Guangze Li, et al. 2022. Towards Video Text

Visual Question Answering: Benchmark and Baseline. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

[56] Minyi Zhao, Miao Wang, Fan Bai, Bingjia Li, Jie Wang, and Shuigeng Zhou. 2022. C3-STISR: Scene Text Image Super-resolution with Triple Clues. In *IJCAI*. 1707–1713.

[57] Shipeng Zhu, Zuoyan Zhao, Pengfei Fang, and Hui Xue. 2023. Improving Scene Text Image Super-Resolution via Dual Prior Modulation Network. *arXiv preprint arXiv:2302.10414* (2023).