Supplementary Material for
Scene Text Telescope: Text-Focused Scene Image Super-Resolution

A. Implementation Details

A.1. Introduction to Several STR Benchmarks

In this section, we introduce several STR benchmarks, including IC13 [4], IC15 [3], and CT80 [8]. These datasets are all available online. Styles of these datasets are quite different from TextZoom [12] and examples of are shown in Figure 1. The detailed introductions for each dataset are as follows:

- **IC13 [4]** contains 1,095 images for validation. Most images are clear and easy to identify by humans.
- **IC15 [3]** contains 1,811 images for validation. The images in this dataset are captured by Google Glasses in natural scenes. Many images are noisy, blurred, and rotated, and some are also of low-resolution by are difficult to recognizer even for humans.
- **CT80 [8]** contains 288 images for validation. Most of the images in this dataset are curved.

We also investigate how many labels in these datasets have appeared in the training set $\mathcal{T}$ of TextZoom. The results are shown in Table 1. We observe that almost half of the labels on average have not been seen during training, which is a great challenge for the super-resolution model.

Dataset	IC13	IC15	CT80
Num. of images	1,095	1,811	288
Num. of labels appears in $\mathcal{T}$	601	881	80
Proportion	54.89%	48.65%	44.44%

Table 1. The proportion of labels that have appeared in $\mathcal{T}$.

Figure 1. Examples of IC13, IC15, and CT80.

A.2. Configuration of TBSRN

In this section, we introduce the details of TBSRN. We first use an STN [2] to rectify the LR image, which is further processed by a CNN layer with the configuration $(k : 3, s : 1, p : 1, i : 3, o : 64)$ to obtain a 64-channel feature map, where k, s, p, i, o denote $kernel$, $stride$, $padding$, $input\ channel$, $output\ channel$, respectively. The feature map goes down to a series of TBSRN blocks, then generates an SR image by pixel shuffling. The two CNN layers follow the configuration $(k : 3, s : 1, p : 1, i : 64, o : 64)$. The generated 64-channel feature map is concatenated with two 32-channel PE so C' is equal to 128. The head number in the Self-Attention Module is set to 4. The size of the hidden layer and output layer in the Position-Wise Feed-Forward Module is set to 128 and 64, respectively. Therefore, the shape of the generated feature map will not change after going through each TBSRN block.

A.3. Configuration of the Pretrained Transformer

We construct the Transformer following other Transformer-based scene text recognizers [10, 13], which contain an encoder and a decoder. The configuration of the encoder is shown in Table 2. Given an input image of size $32 \times 128 \times 3$, the encoder generates a feature map of size $8 \times 32 \times 1024$. Sequentially, the feature map is fed into the decoder. In the decoder, the size of embedding and the size of PE are both set to 512. The head number in the Multi-Head Attention Module (MHA) is set to 4. The size of the hidden layer and the size of the output layer in the Position-Wise Feed-Forward Module are both set to 1024. At each time step, the decoder outputs a 63-dimension vector (10 digits, 26 lowercase letters, 26 uppercase letters, and a stop symbol). We train the Transformer with an Adadelta [14] optimizer and set the learning rate to 1.0. We use a cross-entropy loss to supervise the output text.

A.4. Configuration of VAE

In this section, we introduce the configuration of VAE. As shown in Figure 2, the VAE mainly consists of two parts:

Layer	Configuration	Input Size	Output Size
CNN	(3,1,1,3,64)	32x128x3	32x128x64
Max Pooling	(2,2)	32x128x64	16x64x64
CNN	(3,1,1,64,128)	16x64x64	16x64x128
Basic Block	(1,256)	16x64x128	16x64x256
Max Pooling	(2,2)	16x64x256	16x64x256
Basic Block	(2,256)	8x32x256	8x32x256
Basic Block	(5,512)	8x32x256	8x32x512
Basic Block	(3,1024)	8x32x512	8x32x1024

Table 2. Details of the encoder. The configuration of CNN, Max Pooling, and Basic Block follow the formats ($kernel$, $stride$, $padding$, $input\,channel$, $output\,channel$), ($kernel$, $stride$), and ($block\,number$, $output\,channel$), respectively.

Backbone	SRCNN	SRResNet	TSRN	TBSRN (Ours)
FPS	**253.41**	192.95	40.29	53.14

Table 3. Computational cost of different backbones.

the encoder and the decoder. The image is firstly encoded to a two-dimension vector through the encoder. Then the two-dimension vector is transformed to a 784-dimension vector through a series of linear layers. Finally, the 784-dimension vector is reshaped to an image of size 28×28. The training procedure and the loss functions follow [5].

A.5. Configuration of the Segmentation Model

In this section, we introduce the details of the segmentation model. We propose a variant of U-Net [9] to obtain rough masks for input text images. The configuration is shown in Figure 3. We train the segmentation model with an Adadelta [14] optimizer and set the learning rate to 1.0. We use a cross-entropy loss to supervise the output mask.

B. Additional Experiments

B.1. Computational Cost

In this section, we investigate the computational cost of different backbones to perform this super-resolution task. In the test stage, the Position-Aware Module and the Content-Aware Module can be removed, which will not bring additional time overhead. We calculate FPS for SRCNN [1], SRResNet [6], TSRN [12], and the proposed TBSRN. All the FPSs are tested on four TITAN Xp GPUs with a single image. The results are shown in Table 3. Even though SRCNN and SRResNet are faster than our model, the two backbones do not take sequential information into consideration, which is not suitable for tackling text images. Moreover, the proposed TBSRN is faster than TSRN benefited from the parallelism of Transformer [11]. Therefore, the proposed TBSRN shows great superiority in the trade-off between speed and accuracy.

$\mathcal{L}_{\text{PSM}}$	$\mathcal{L}_{\text{POS}}$	Accuracy			
		Easy	Medium	Hard	Average
L1	L1	57.32%	44.08%	34.10%	45.92%
L1	L2	54.97%	40.75%	32.39%	43.48%
L2	L1	**59.60%**	**47.10%**	**35.30%**	**48.10%**
L2	L2	57.01%	42.45%	33.36%	45.11%

Table 4. Choices between the L1 loss and the L2 loss. We conduct this experiment on TextZoom. The accuracy is tested on CRNN.

B.2. Choices Between the L1 Loss and the L2 Loss

We conduct several experiments on whether the L1 loss or the L2 loss is suitable for $\mathcal{L}_{\text{PSM}}$ and $\mathcal{L}_{\text{POS}}$. The results are shown in Table 4. We observe that the accuracy achieves the best when $\mathcal{L}_{\text{PSM}}$ chooses the L2 loss and $\mathcal{L}_{\text{POS}}$ chooses the L1 loss. Since $\mathcal{L}_{\text{POS}}$ is usually a small value (*i.e.* very close to 0.001), it is hard for the model to converge if we choose the L2 loss for $\mathcal{L}_{\text{POS}}$ due to the small gradient.

B.3. Results on More Robust Recognizers

In this section, we employ several more robust recognizers to validate whether the preprocessor also works on these recognizers. Specifically, we leverage NRTR [10][1], SEED [7][2], as well as AutoSTR [15][3], all of which are available on Github in terms of code and pre-trained models. The results are shown in Table 5. Through the results, we observe that the Position-Aware Module and Content-Aware Module also work on robust recognizers, while the proposed TBSRN shows close performance compared with TSRN [12].

C. Visualization

C.1. Visualization for the Effect of $\mathcal{L}_{\text{PSM}}$

In Figure 4, we show the visualization of examples which are generated in the absence of $\mathcal{L}_{\text{PSM}}$.

C.2. Visualization for the Position-Aware Module

In this section, we display some visualizations for the Position-Aware Module. Examples are shown in Figure 5. For the SR images generated by the model with $\mathcal{L}_{\text{POS}}$, the boundary between characters is clearer, which is helpful for the following recognition task.

C.3. Visualization for the Content-Aware Module

In this section, we display some visualizations for the Content-Aware Module. Examples are shown in Figure 6. For the SR images generated by the model with $\mathcal{L}_{\text{CON}}$, the characters are clearer. Furthermore, the model without $\mathcal{L}_{\text{CON}}$ is weak in handling those confusable characters (*e.g.* “c” and “o”, “t” and “l”).

[1] https://github.com/Belval/NRTR
[2] https://github.com/Pay20Y/SEED
[3] https://github.com/AutoML-4Paradigm/AutoSTR

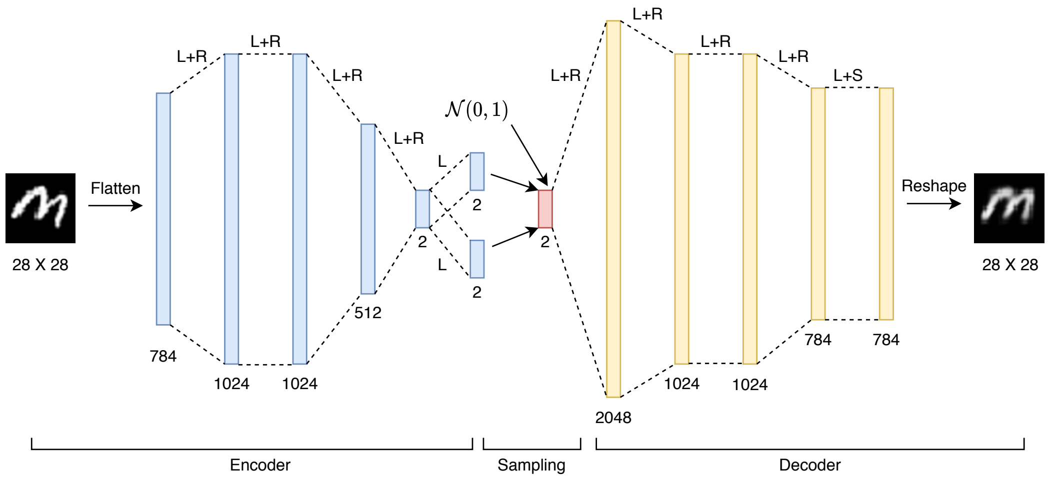

Figure 2. The illustration of VAE. “L”, “R”, “S” denote a linear layer, a ReLU activation layer, a Sigmoid activation layer.

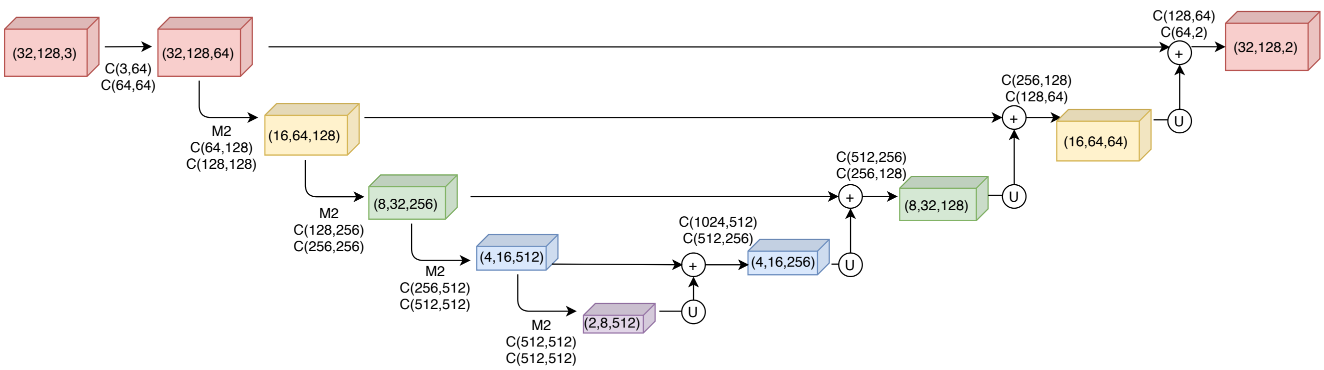

Figure 3. The illustration of the segmentation model. C(x, y) means a CNN with $input\ channel = x$ and $output\ channel = y$. $kernel$, $stride$, and $padding$ of CNN are set to 3, 1, and 1. “M2” denotes a Max Pooling layer with $kernel = 2$ and $stride = 2$. “U” stands for a 2×2 upsampling operation with the bilinear interpolation. “+” means the concatenation operation. We display the size of each feature map in the center of each feature map following the format ($height$, $width$, $channel$).

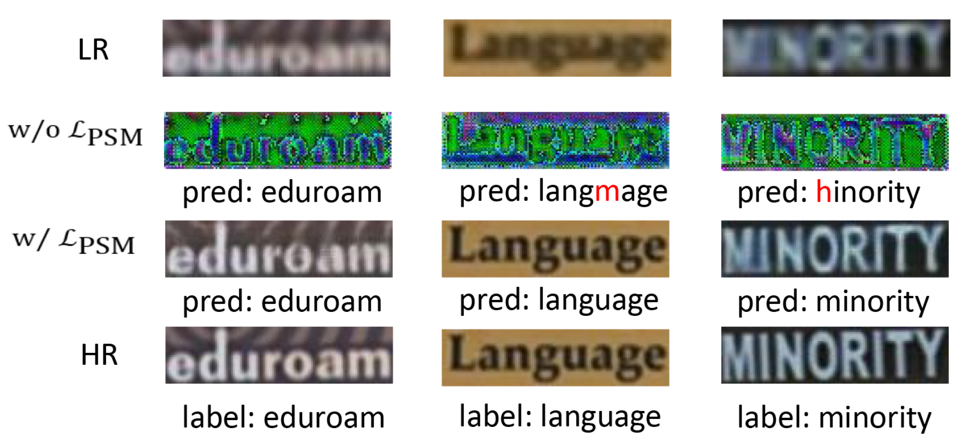

Figure 4. Visualization for effect of $\mathcal{L}_{\text{PSM}}$.

Backbone	$\mathcal{L}_{POS}$ & $\mathcal{L}_{CON}$	NRTR [10]				SEED [7]				AutoSTR [15]			
		Easy	Medium	Hard	Average	Easy	Medium	Hard	Average	Easy	Medium	Hard	Average
BICUBIC	-	65.1%	46.2%	31.8%	48.8%	70.2%	48.1%	34.8%	52.2%	67.8%	46.2%	32.9%	50.1%
SRCNN [1]	-	60.0%	39.3%	28.6%	43.7%	71.5%	44.4%	32.3%	50.7%	69.8%	43.6%	31.6%	49.6%
	✓	59.1%	39.5%	28.6%	43.4%	70.3%	46.4%	33.4%	51.3%	67.9%	44.8%	31.9%	49.4%
SRResNet [6]	-	60.1%	48.8%	33.4%	48.3%	70.2%	56.8%	38.5%	56.1%	70.7%	55.1%	37.2%	55.4%
	✓	63.1%	49.6%	34.8%	50.1%	73.2%	58.0%	39.5%	57.9%	73.3%	57.2%	39.7%	57.8%
TSRN [12]	-	64.4%	50.0%	34.9%	50.7%	74.8%	56.3%	39.2%	57.9%	75.2%	56.1%	39.7%	58.1%
	✓	66.8%	**53.6%**	**37.3%**	**53.5%**	75.8%	**60.1%**	**41.0%**	**60.0%**	75.7%	59.2%	40.2%	59.5%
TBSRN	-	65.3%	49.0%	37.2%	51.4%	74.6%	56.2%	41.8%	58.6%	74.6%	55.1%	41.5%	58.1%
	✓	**67.6%**	52.2%	37.1%	53.3%	**76.8%**	58.3%	40.4%	59.7%	**76.8%**	**59.5%**	**41.8%**	**60.5%**

Table 5. The experimental results of TextZoom on more robust recognizers.

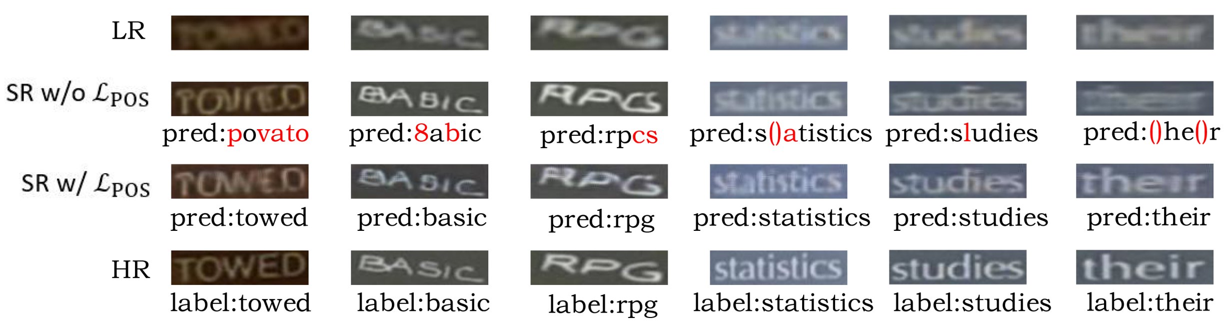

Figure 5. Visualization for the Position-Aware Module.

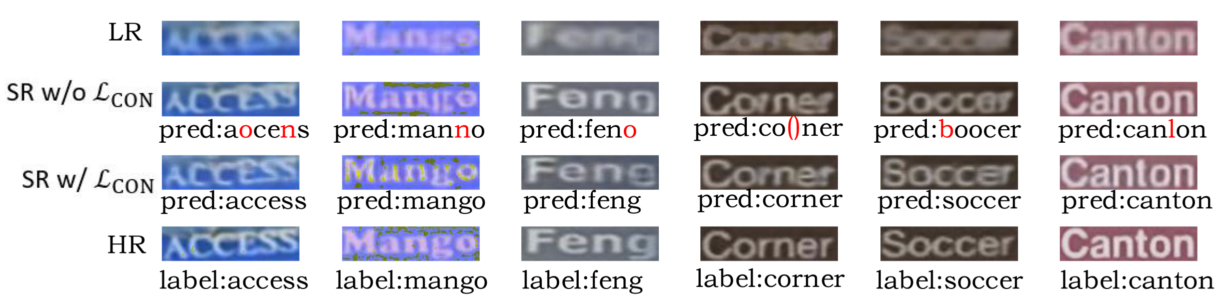

Figure 6. Visualization for the Content-Aware Module.

References

[1] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. In *European conference on computer vision*, pages 184–199. Springer, 2014. 2, 4

[2] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in neural information processing systems*, pages 2017–2025, 2015. 1

[3] Dimosthenis Karatzas, Lluis Gomez-Bigorda, Anguelos Nicolaou, Suman Ghosh, Andrew Bagdanov, Masakazu Iwamura, Jiri Matas, Lukas Neumann, Vijay Ramaseshan Chandrasekhar, Shijian Lu, et al. Icdar 2015 competition on robust reading. In *2015 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1156–1160. IEEE, 2015. 1

[4] Dimosthenis Karatzas, Faisal Shafait, Seiichi Uchida, Masakazu Iwamura, Lluis Gomez i Bigorda, Sergi Robles Mestre, Joan Mas, David Fernandez Mota, Jon Almazan Almazan, and Lluis Pere De Las Heras. Icdar 2013 robust reading competition. In *2013 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1484–1493. IEEE, 2013. 1

[5] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 2

[6] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4681–4690, 2017. 2, 4

[7] Zhi Qiao, Yu Zhou, Dongbao Yang, Yucan Zhou, and Weiping Wang. Seed: Semantics enhanced encoder-decoder framework for scene text recognition. In *CVPR*, pages 13528–13537, 2020. 2, 4

[8] Anhar Risnumawan, Palaiahankote Shivakumara, Chee Seng Chan, and Chew Lim Tan. A robust arbitrary text detection system for natural scene images. *Expert Systems with Applications*, 41(18):8027–8048, 2014. 1

[9] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 2

[10] Fenfen Sheng, Zhineng Chen, and Bo Xu. Nrtr: A no-recurrence sequence-to-sequence model for scene text recognition. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 781–786. IEEE, 2019. 1, 2, 4

[11] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 2

[12] Wenjia Wang, Enze Xie, Xuebo Liu, Wenhai Wang, Ding Liang, Chunhua Shen, and Xiang Bai. Scene text image super-resolution in the wild. *European conference on computer vision*, 2020. 1, 2, 4

[13] Lu Yang, Peng Wang, Hui Li, Zhen Li, and Yanning Zhang. A holistic representation guided attention network for scene text recognition. *Neurocomputing*, 414:67–75, 2020. 1

[14] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012. 1, 2

[15] Hui Zhang, Quanming Yao, Mingkun Yang, Yongchao Xu, and Xiang Bai. Efficient backbone search for scene text recognition. In *ECCV*, 2020. 2, 4