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 recognize 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 $T$ 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.

![Table 1. The proportion of labels that have appeared in $T$.](image)

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 (MHA) is set to 4. 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:
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), and (block number, output channel), respectively.

<table>
<thead>
<tr>
<th>Backbone</th>
<th>SRCNN</th>
<th>SRResNet</th>
<th>TSRN</th>
<th>TBSRN (Ours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>FPS</td>
<td>253.41</td>
<td>192.95</td>
<td>40.29</td>
<td>53.14</td>
</tr>
</tbody>
</table>

Table 3. Computational cost of different backbones.

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.

B. Additional Experiments

B.1. Computational Cost

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 $L_{\text{PSM}}$, the boundary between characters is clearer, which is helpful for the following recognition task.

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 $L_{\text{PSM}}$ and $L_{\text{POS}}$. The results are shown in Table 4. We observe that the accuracy achieves the best when $L_{\text{PSM}}$ chooses the L2 loss and $L_{\text{POS}}$ chooses the L1 loss. Since $L_{\text{POS}}$ 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 $L_{\text{POS}}$ 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] and AutoSTR [15], as well as AutoSTR [15], 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 $L_{\text{PSM}}$

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

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 $L_{\text{POS}}$, 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 $L_{\text{CON}}$, the characters are clearer. Furthermore, the model without $L_{\text{CON}}$ is weak in handling those confusable characters (e.g. “c” and “o”, “t” and “I”).

<table>
<thead>
<tr>
<th>$L_{\text{PSM}}$</th>
<th>$L_{\text{POS}}$</th>
<th>Easy</th>
<th>Medium</th>
<th>Hard</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1</td>
<td>L1</td>
<td>57.32%</td>
<td>44.08%</td>
<td>34.10%</td>
<td>45.92%</td>
</tr>
<tr>
<td>L1</td>
<td>L2</td>
<td>54.97%</td>
<td>40.75%</td>
<td>32.39%</td>
<td>43.48%</td>
</tr>
<tr>
<td>L2</td>
<td>L1</td>
<td>59.60%</td>
<td>47.10%</td>
<td>35.30%</td>
<td>48.10%</td>
</tr>
<tr>
<td>L2</td>
<td>L2</td>
<td>57.01%</td>
<td>42.45%</td>
<td>33.36%</td>
<td>45.11%</td>
</tr>
</tbody>
</table>

1https://github.com/Belval/NRTR
2https://github.com/Pay207/SEED
3https://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 \( 2 \times 2 \) 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 \( L_{PSM} \).
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Easy</td>
<td>Medium</td>
<td>Hard</td>
<td>Average</td>
</tr>
<tr>
<td>BICUBIC</td>
<td>-</td>
<td>65.1%</td>
<td>46.2%</td>
<td>31.8%</td>
</tr>
<tr>
<td>SRCNN [1]</td>
<td>-</td>
<td>60.0%</td>
<td>39.3%</td>
<td>28.6%</td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td>59.1%</td>
<td>39.5%</td>
<td>28.6%</td>
</tr>
<tr>
<td>SRCResNet [6]</td>
<td>-</td>
<td>60.1%</td>
<td>48.8%</td>
<td>33.4%</td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td>63.1%</td>
<td>49.6%</td>
<td>34.8%</td>
</tr>
<tr>
<td>TSRN [12]</td>
<td>-</td>
<td>64.4%</td>
<td>50.0%</td>
<td>34.9%</td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td>66.8%</td>
<td>53.6%</td>
<td>37.3%</td>
</tr>
<tr>
<td>TBSRN</td>
<td>-</td>
<td>65.3%</td>
<td>49.0%</td>
<td>37.2%</td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td>67.6%</td>
<td>52.2%</td>
<td>37.1%</td>
</tr>
</tbody>
</table>

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


