TextSSR: Diffusion-based Data Synthesis for Scene Text Recognition

Supplementary Material

6. More Implementation Details

6.1. Model specific settings

In the setup of Sec. 3.2, we consider the specific characteristics of scene text: L is set to 25, and P_M is configured to 128, assuming a background color of 255. It is worth noting that our method is not limited to this configuration. This configuration theoretically enables the generation of text up to 255 characters in length, allowing for background color selection from 255 and character values from 0 to 254. Each character image is set to a 64×64 square, resulting in a character glyph image of size 25×64×64. The deformed ViT is configured with a patch size of 8, generating a latent feature vector of size 65×1024, where 1024 represents the dimensionality of the control information required by the CDM.

After passing through the VAE in Sec. 3.3, the dimensions of outputs are uniformly [4, 16, 16], while Z_M is formatted as [1, 16, 16]. These are concatenated to form [13, 16, 16]. Therefore, the Conv2d Layer has an input dimension of 13 and an output dimension of 4, producing an output of [4, 16, 16] to match the original input dimensions of the U-Net.

To meet the rendering requirements for visible characters in the majority of languages, we utilize to employ Puhui Font, an open-source and commercially-free font tool. It adheres to the latest Chinese national standard, GB18030-2022, and supports 178 languages.

6.2. More Details for Datasets

We will further elaborate on the data processing related to training and generation.

Training Data. To train our generative model, we utilize the large-scale multilingual text image dataset, AnyWord-3M [41]. It contains real annotated text boxes and text contents, designed for scene text detection and recognition tasks, which we collectively call AnyWord-Scene. This collection includes a range of popular datasets such as ArT [6], COCO-Text [42], RCTW [37], LSVT [40], MLT [28], MTWI [16], and ReCTS [55]. In addition, two larger datasets are included: AnyWord-Wukong [14] and AnyWord-Laion [34], which provide a large collection of images with bounding boxes and text content obtained by the PP-OCRv3 [23] detection and recognition model. We filter out anomalous images with pure white backgrounds from the AnyWord-3M dataset. Ensuring that when cropping local images, we minimize the inclusion of white borders, thereby reflecting real-world conditions. The processed AnyWord-Wukong and AnyWord-Laion datasets together contain a total of 3,430,412 complete images, from which we crop 14,856,392 local regions containing text instances for the first training stage. In the second training stage, we utilize 78,395 full images from processed AnyWord-Scene, cropping 201,599 text regions from these.

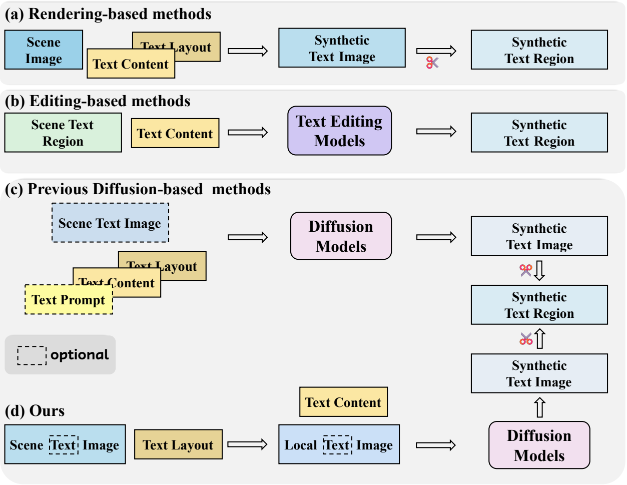


Figure 7. The pipeline comparison between TextSSR and previous methods. In (c), "optional" indicates that either or both options should be provided. And in (d), it means that the base image can be used with or without text.

Computational Overhead and Runtime Efficiency. The training times for our three stages (VAE fine-tuning, UNet pretraining, and UNet fine-tuning) are 192, 400, and 200 GPU-hours, respectively. For inference, using a single RTX-3090 GPU with diffusion_steps=20 and batch_size=32, 5k batches take 26 hours, averaging 0.59 seconds per image.

Data for Accuracy Evaluation. Meanwhile, in Accuracy Evaluation we employ datasets that the model has not previously encountered. They represent a range of image difficulty levels and cover multiple languages. Specifically, we use the following datasets for evaluation:

- **IC13** [21]: This benchmark is designed for relatively regular text detection and recognition. We use 233 full images and 917 cropped text images from the test set for evaluation.
- **IC15** [22]: This dataset contains more challenging real-world scene text, derived from incidental scene captures where the text was not the primary focus. We employ 500 full images and 2,077 cropped text images for evaluation.
- **Shopsign** [54]: This dataset consists of Chinese scene

Method	French			German			Japanese			Traditional Chinese		
	SeqAcc(%)↑	NED↑	FID-R↓	SeqAcc(%)↑	NED↑	FID-R↓	SeqAcc(%)↑	NED↑	FID-R↓	SeqAcc(%)↑	NED↑	FID-R↓
TextDiffuser	12	0.2953	231.47	5	0.1707	240.55	0	0.0014	280.95	0	0.0011	259.46
GlyphControl	0	0.0188	268.63	0	0.0264	277.10	0	0.0000	276.20	0	0.0011	261.79
AnyText	24	0.4508	181.22	16	0.3276	198.34	3	0.1696	215.89	21	0.3557	208.85
TextDiffuser2	9	0.2891	169.56	6	0.2178	208.04	0	0.0015	230.02	0	0.0080	190.93
TextSSR	**68**	**0.8024**	**107.78**	**75**	**0.8734**	**102.86**	**24**	**0.5545**	**127.87**	**55**	**0.7304**	**112.81**
Real	94	0.9858	0	94	0.9810	0	83	0.9562	0	98	0.9967	0

Table 6. Quantitative comparison of multilingual text image generation methods. For each language, 100 images are used for test.

text, primarily from shop signs. We selecte 183 full images and 932 cropped text images from this dataset.

Base Data for Scalable Generation. We utilize the TextOCR [39], a large-scale scene text dataset including 25,119 images, as the base images for synthetic data generation. To mimic realistic conditions where unlabeled data is abundant, we use pseudo-labels generated by the PP-OCRv3 model, thus simulating a scenario without human annotation.

Generation of TextSSR-F. After the previous step, we obtain 188,526 text regions annotated by the PP-OCRv3 [23] model. Based on the anagram method (described in Section 3.4), we expand the data and filter with the SVTRv2 [11] model, yielding a final dataset of 3,551,396 fully usable text instances, referred to as TextSSR-F.

Quality Filtering Bias Our filtering process employs a double-check mechanism: given the label the generation model attempts to generate, and SVTRv2 is used to verify that the recognized text matches the label. An error occurs only if both the generation model and SVTRv2 fail simultaneously. This pipeline ensures the correctness of most TextSSR-F instances. To validate this, we randomly sample 300 instances from TextSSR-F and recruit three assessors each checking 100 instances. The average accuracy is 98.67% (98%, 98%, 100%).

Impact of Pseudo-Labeling When OCR pseudo-label errors occur, the generation process still attempts to follow the pseudo-labels (see examples in Fig. 8), so the side impact is relatively limited.

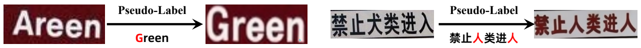


Figure 8. Examples of correct rendering despite incorrect English and Chinese pseudo-labels.

6.3. Training and Evaluation Details

6.3.1. Training Details

We fine-tune our generative model using Stable Diffusion 2.1 (SD 2-1) [33] on eight NVIDIA 3090 GPUs. First, we train the VAE on the full AnyWord dataset with a total batch size of 512 and 256x256 image patches for 150k steps. Then, we freeze the VAE and train the CDM in two stages: 50k steps on AnyWord-Wukong and AnyWord-Laion datasets to pre-train, followed by 25k steps on the AnyWord-Scene dataset to fine-tune, using a total batch size of 256.

6.3.2. Accuracy Evaluation Details

For a fair comparison in our accuracy evaluation, we render all visible bounding boxes and contents annotated in the test datasets. In cases where certain models could not render longer texts or handle multiple text instances per image, we will restrict the input information to within their acceptable ranges, while padding the missing portions. Our model is also limited, with only the first 25 characters rendered for single-character features. For all models, the number of timesteps in the sampling process is set to 20. The evaluation code for generated results is based on the open-source evaluation scripts from AnyText [41] and UDiffText [57]. Except for GlyphControl [50], which requires additional image descriptions to function properly, the other methods only use their predefined text prompts.

6.3.3. Expanded Multilingual Evaluation

We have added four languages (French, German, Japanese, and Traditional Chinese) and use the multilingual version of SVTRv2 for evaluation (see Tab. 6). We also provide illustrative examples in Fig. 9 to validate the generalization to non-English languages. TextSSR generates correct instances while others mostly fail.

6.3.4. Realism and Scalability Evaluation Details

In the Realism and extended experiments, CRNN is trained [1] with a batch size of 64 on a single 3090 GPU for 10k steps. The data augmentation configurations preset in the codebase are utilized throughout the training process.

6.3.5. Usability Assessment Details

In the Usability experiments, we train two widely-used STR models—CRNN [36], and MAERec [20]—on the generated data to assess the effectiveness of our synthetic data in enhancing STR performance. All models are trained using the OpenOCR framework, with a total batch size of 1024 on four 3090 GPUs for 20 epochs.

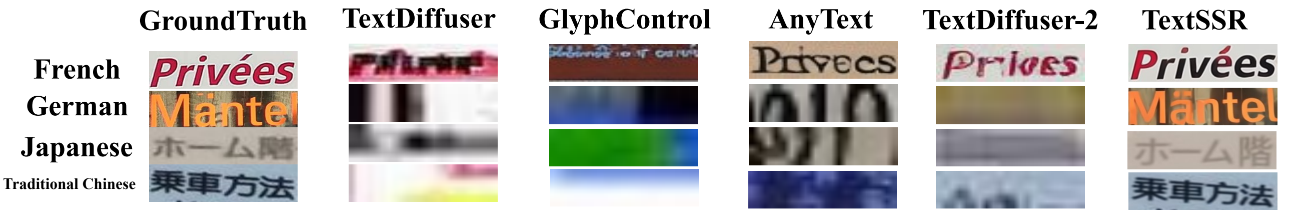
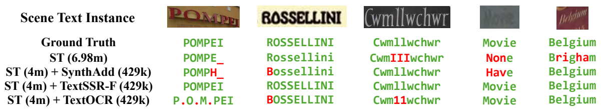


Figure 9. Visualization of synthesized multilingual examples.

To vividly demonstrate that TextSSR significantly enhances the performance of STR models under challenging scenarios, we conduct a small-scale validation experiment. In this experiment, we limit the dataset size to 429k and employ an identical NRTR [35] model trained under the same configuration for comparison. The experimental results indicate that the model trained on TextSSR-F exhibits more realistic performance when dealing with challenging text conditions such as perspective distortion and blurring. We provide visual comparisons in Fig. 10, showcasing TextSSR's superior performance in recognizing low-resolution and perspective-distorted text.



Scene Text Instance					
Ground Truth	POMPEI	ROSSELLINI	Cwmllwchwr	Movie	Belgium
ST (6.98m)	POMPE_	Rossellini	CwmIIIwchwr	None	Brigham
ST (4m) + SynthAdd (429k)	POMPH_	Bossellini	Cwmllwchwr	Have	Belgium
ST (4m) + TextSSR-F (429k)	POMPEI	ROSSELLINI	Cwmllwchwr	Movie	Belgium
ST (4m) + TextOCR (429k)	P.O.M.PEI	BOSSELLINI	Cwm11wchwr	Movie	Belgium

Figure 10. Visualization of recognition results on NRTR.

6.3.6. Ablation Study Details

The ablation study can be considered a simplified version of the second stage of training, with all settings kept consistent except for the reduction of training steps to 5k. To align with the full-image inference process used in other methods, the image size is set to 512×512, although training is conducted at a resolution of 256. The "Char-Glyph" ablation experiment involves removing the condition from the CDM training, while the "Char-Position" ablation renders all characters uniformly at a pixel value of 127.

7. Visualization

7.1. Visualization Analysis

Fig. 4 sequentially simulates various situations, including English text in a regular scene, text under challenging conditions, and Chinese text in a natural setting. TextSSR consistently generates accurate and high-quality visual text, demonstrating several powerful capabilities: (1) it can synthesize arbitrary text with standard glyphs from any language, as shown in examples of both Chinese and English; (2) it learns font style information from surrounding context, such as the font color in Sample 1, which is derived from the horizontal line below; (3) it synthesizes correct text even without strong background information, as illustrated in Sample 2, where the local image provides no usable information for imitation; and (4) it exhibits scale invariance, allowing for text synthesis in scenes of any size, with the three samples representing large, small, and medium text sizes, respectively.

7.2. Function Demonstration Platform

We have concretely implemented the inference process and build a demonstration demo. To ensure that the user input matches the label format used during training, we recalculate text boxes that align with the input text location after the user applies the mask. As shown in Fig. 12, the text is roughly displayed within the user-specified area, though it does not follow the mask strictly.

7.3. Ablation Visualization Results

Fig. 13 and Fig. 11 illustrate the impact of character-level position and glyph on the rendering results of TextSSR. The results indicate that omitting either component leads to issues such as character deformation, incorrect characters, and duplication errors in some cases, further supporting the findings of the ablation study.

7.4. More Visualization Results

To provide a detailed illustration of the synthesis process and effectiveness of TextSSR, Fig. 14 illustrates TextSSR's synthesis process, showing how it reconstructs local images from original regions and crops them to obtain final results. Comparisons with ground truth demonstrate TextSSR's strong synthesis capabilities across diverse scenarios, including regular text, low-resolution text, curved text, perspective text, multilingual text and multi-oriented text.

7.5. Failure Cases

It is important to note that our synthesis method is not flawless and has certain limitations. Fig. 15 presents several common failure cases, which can be attributed to the following reasons:

1. **Long text:** Excessively long text can confuse the model, resulting in disordered text images. This issue is exac-

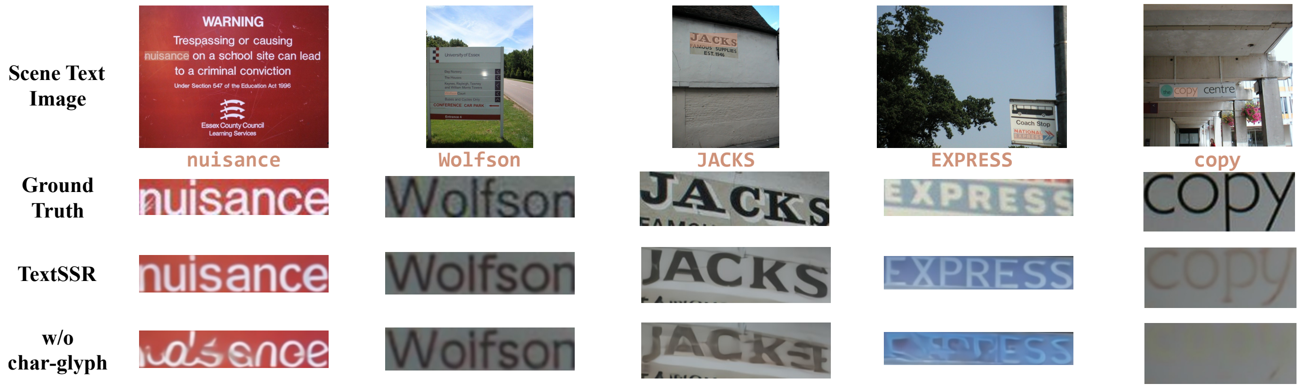
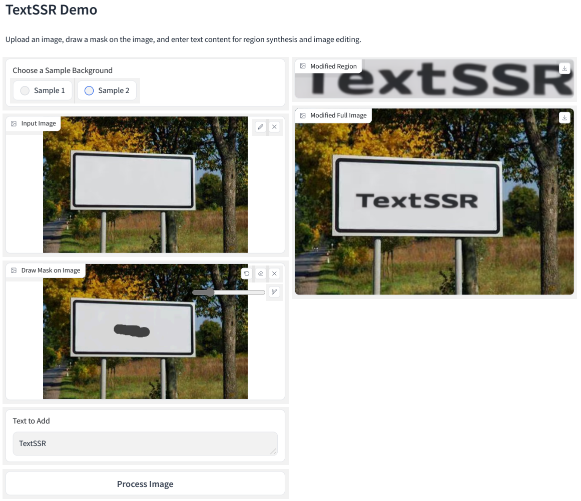


Figure 11. Visualization results of TextSSR with and without character-level glyph prior.



TextSSR Demo

Upload an image, draw a mask on the image, and enter text content for region synthesis and image editing.

Text to Add: TextSSR

Figure 12. Users will select a scene image as the base, perform mask marking in the designated area, and then input the text content to be written. After processing, the desired text region and the edited original image will be obtained.

erbated by the limited amount of training data for such cases.

2. **Blurred regions:** When the text region itself is excessively blurred, the model struggles to accurately reconstruct and synthesize the text.
3. **Multi-directional text:** The model, primarily trained on horizontally aligned text, faces challenges with multi-directional text, especially vertical text. Applying rotation-based post-processing, as used in STR methods, could be a potential solution.
4. **Incorrect text labels:** Errors in manual labeling can lead to mismatches between the rendered regions and their corresponding labels.
5. **Language Characteristics:** The performance on Chinese text is generally worse than on English, due to the higher number of characters and the complexity of Chinese characters.

Despite the minority in quantity, handling challenging text instances are also important. We plan to tackle these instances as follows: splitting long text into shorter segments, simulating blur by adding noise and augmenting multi-directional text via rotation based on common instances, leveraging render-based data for pretraining on multilingual characters, etc.

8. Discussion

However, our study has limitations and avenues for further research, including the following: (1) The text location and the text content must be paired. While we utilize the anagram-based method to mitigate this issue, we will design methods for reasonable, large-scale usable pairings for broader synthesis considerations. (2) Currently, large-scale synthetic post-processing relies on an STR model; we aim to integrate a self-checking mechanism into the entire framework to verify the correctness of the synthesized output. This could further enhance learning and adjust the arrangement of text location until generating usable text correctly. (3) Due to available large-scale scene text images already used for training, we plan to collect a larger dataset of untrained text images for the base images, creating a more extensive synthetic dataset to benefit the STR community. (4) Although the generation is related to the surrounding context, currently TextSSR does not fully address this issue due to lack of customized design. However, by substituting certain TextSSR component, e.g., the anagram expansion, or using LLM to recommend contextually appropriate content, TextSSR can largely alleviate it while the rest TextSSR components can still be reused. We will improve the semantic diversity and contextual realism from these aspects in future. (5) While our primary focus is on STR, our approach can also benefit other downstream tasks. For example, by

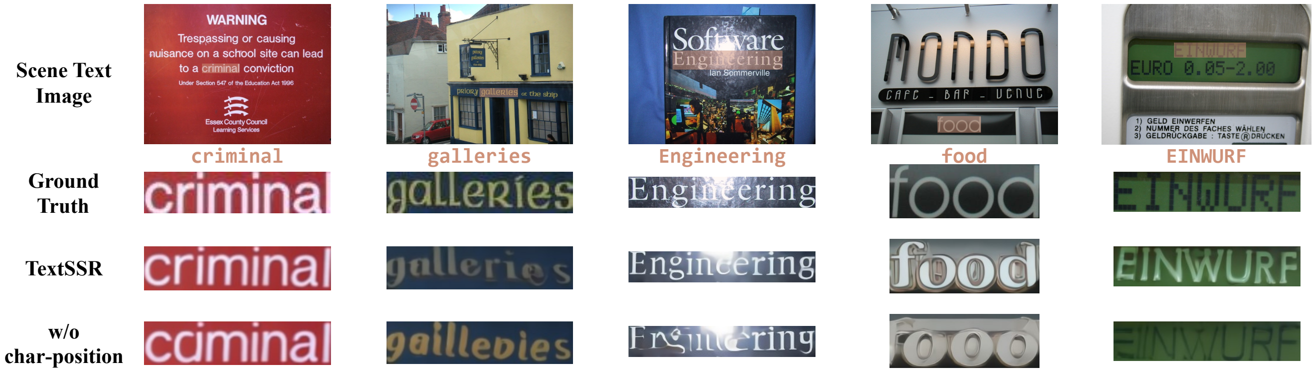


Figure 13. Visualization results of TextSSR with and without character-level position prior.

directly writing text onto the background or editing original text, our method can generate new data for text detection and document understanding tasks. We also agree that domain generalization is a valuable topic. We will investigate other downstream applications and discuss broader specialized domains in future.

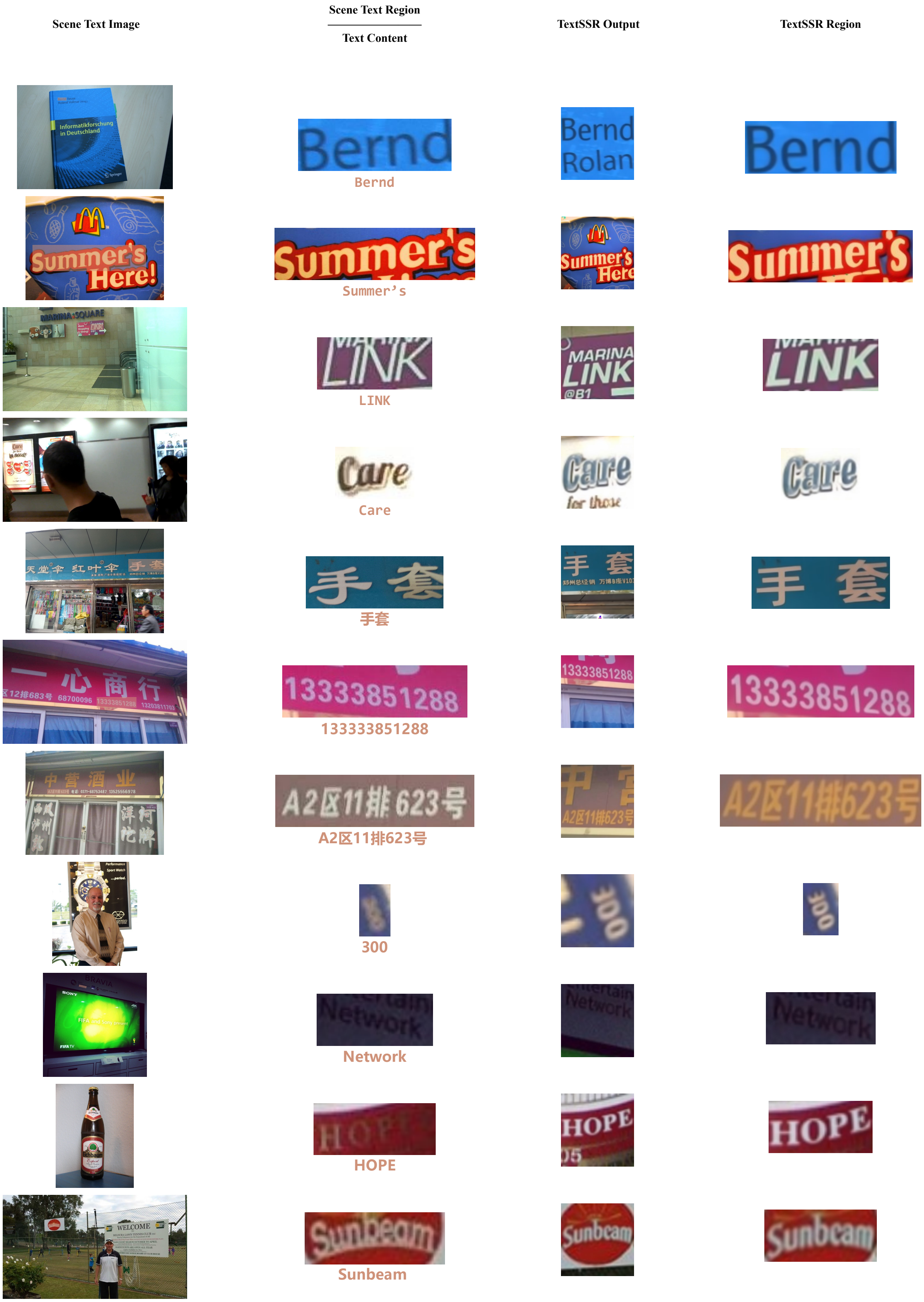


Figure 14. More visualization results for TextSSR.

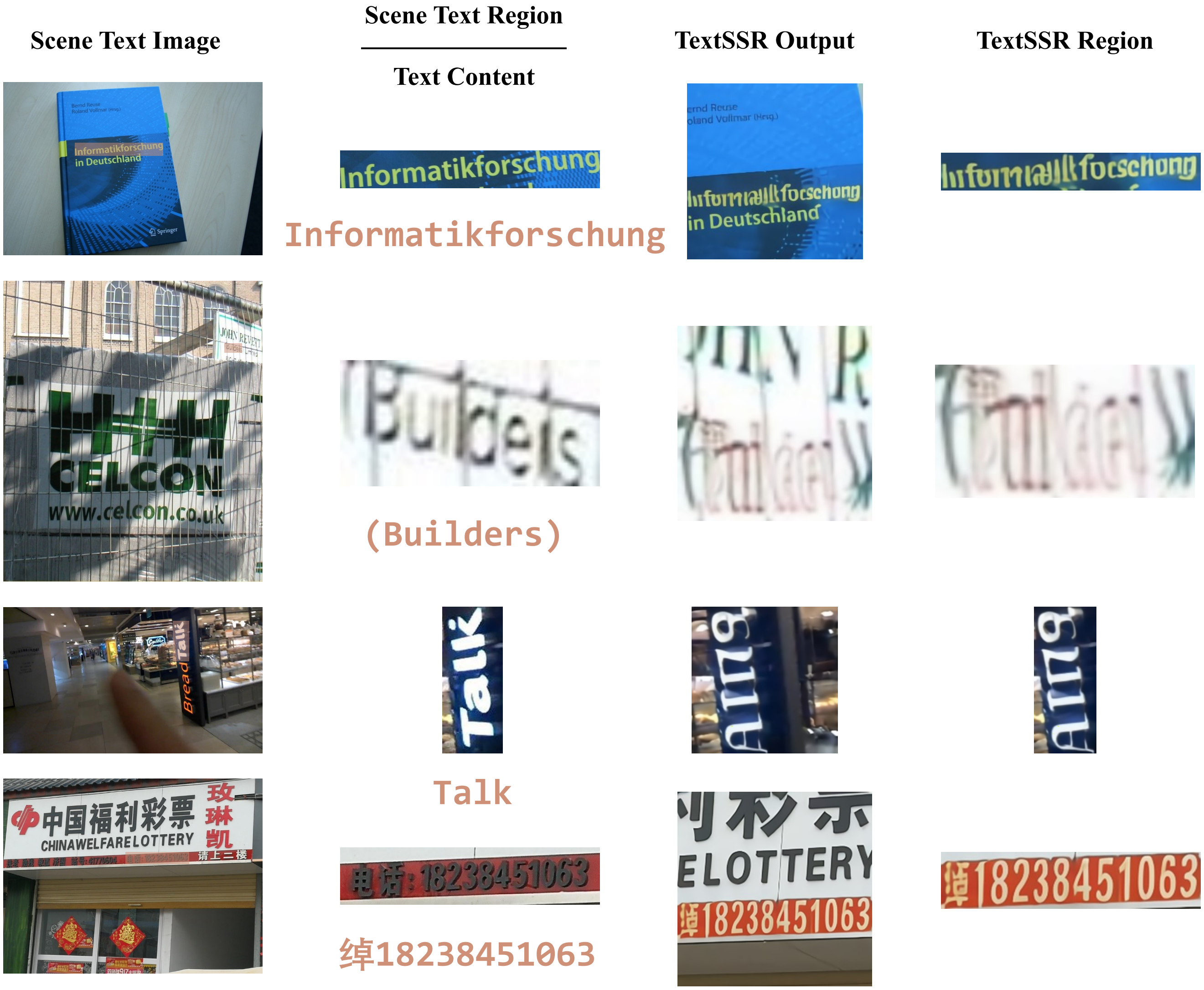


Figure 15. Failure Cases. We show several disappointing synthesis results produced by TextSSR.