RewriteNet: Reliable Scene Text Editing with Implicit Decomposition of Text Contents and Styles

Junyeop Lee[1*], Yoonsik Kim[2*]
Seonghyeon Kim[2], Moonbin Yim[2], Seung Shin[2], Gayoung Lee[2], Sungrae Park[1†]

[1]Upstage AI Research, Upstage
[2]Clova AI Research, NAVER Corp.
{junyeop.lee, sungrae.park}@upstage.ai
{yoonsik.kim90, kim.seonghyeon, moonbin.yim, seung.shin, gayoung.lee}@navercorp.com

Abstract

Scene text editing (STE), which converts a text in a scene image into the desired text while preserving an original style, is a challenging task due to a complex intervention between text and style. In this paper, we propose a novel STE model, referred to as RewriteNet, that decomposes text images into content and style features and re-writes a text in the original image. Specifically, RewriteNet implicitly distinguishes the content from the style by introducing scene text recognition. Additionally, independent of the exact supervisions with synthetic examples, we propose a self-supervised training scheme for unlabeled real-world images, which bridges the domain gap between synthetic and real data. Our experiments present that RewriteNet achieves better generation performances than other comparisons. Further analysis proves the feature decomposition of RewriteNet and demonstrates the reliability and robustness through diverse experiments. Our implementation is publicly available at https://github.com/clova ai/rewritenet

1. Introduction

Scene text editing (STE) is a task of image synthesis that replaces the text in a scene image to the desired text while preserving a style such as a font type, font size, text alignment, and background. As shown in Figure 1 (a), the texts in the image patches are converted while keeping the original styles. As a core technology for virtual reality, STE can be employed for scene text images to replace the text contents (e.g. Figure 1 (b) and (c)). As can be seen, since real-world text images have complex backgrounds and text styles, STE methods should address intricately intertwined tasks including image in-painting, style extraction, charac-

(a) Original and edited text images.

(b) Original scene images.

(c) Edited scene images.

Figure 1. Examples of STE results. (a) is the original text images (leftmost) and the text edited images where target texts are "Abstract", "CLOSE", and "Domain". (b) and (c) show real-world applications with original and edited scene images. All results are generated by RewriteNet.

* indicates equal contribution.
† indicates corresponding author.

ter rendering, and text localization.

Previous STE methods [30, 40, 42] follow a framework with two stages: text deletion and text conversion. Text deletion module generates text erased background, which can be thought of as an image in-painting task specialized for scene text images [26, 36, 41, 43]. Text conversion module renders the desired text where the text-related styles in the original image are transferred, and then, two outputs generated from text deletion and conversion module are harmonically fused. By incorporating the text deletion, previous methods show the feasibility of STE. However, since the text deletion heavily depends on visual features when distinguishing between the text region and background region, it causes two limitations on the two-stage STE methods. First, the text deletion has never utilized text information, which could be a key for understanding scene text images. Second, the text deletion module cannot learn from real-world examples since requiring visual supervision for text-erased backgrounds.

In this paper, we present a novel representational learning-based STE framework, referred to as RewriteNet, which implicitly distinguishes content and style features without the explicit text deletion tasks. Specifically, we introduce a scene text recognition (STR) module to disentangle content features representing a series of characters from style features containing anything others such as font style, font color, text alignment, and background. In addition, to avoid mixing other visual information with content features, we detach the gradient flow from the final generation to the content features. With separately extracted style and content features, a generator can be trained to synthesize an image with a target text while preserving the style of the original image. Thus, RewriteNet replaces the text deletion and conversion stages of previous work with a simple encoder in the latent space and the model can be trained in an end-to-end manner.

We also propose a self-supervised training scheme that does not require additional annotation cost and enables to exploit unlabeled real-world images. The proposed self-supervised training scheme prevents the trained model to be biased in synthetic styles and bridges the domain gap between training and test environments. As shown in Figure 1, our model robustly generates text-edited images where the styles of original images are well preserved. Our extensive experiments demonstrate the superiority of RewriteNet and further analysis shows that our method reliably decomposes content and style features.

2. Related Works

2.1. Scene Text Editing

As the growth of generation model [44, 45], STE has been actively studied for its various applications. Previous STE methods mainly have proposed multiple sub-modules to extract a background and spatial text alignment, and a single fusion module to generate a text-edited image with the identified information. Specifically, initial STE work [30] segments binary mask for each character and switches it into the desired character. Although it has shown that their character correction method can be applied in real-world images, it cannot deal with different lengths between the text in the original image and the desired text. Moreover, its simple rule-based segmentation module could critically affect the generation performance.

Recently, Wu *et al*. [40] and Yang *et al*. [42] have proposed word-level STE methods using text background inpainting and fusion modules. These methods attempt to train the model to separate the text region and the background region using the text-erased image. They could successfully conduct word-level STE, however, sometimes they fail to edit the text of the complex style images. To deal with this problem, we exploit text information and real-world images by proposing a representational learning-based word-level STE framework.

2.2. Image-to-Image Translation

Image-to-image translation methods have been widely researched due to their practical usages. Isola *et al*. [13] have proposed paired image-to-image translation with conditional GAN to learn a mapping from the input image to the output image. To address unpaired image-to-image translation, UNIT [19] and MUNIT [11] assume fully and partially shared latent space, respectively. Ma *et al*. [22] have proposed an exemplar guided image translation method with a semantic feature mask that does not require additional labels for feature masks. Motivated by previous works, we introduce partially shared latent space assumption and the self-supervised training scheme with STE specialized proposals.

3. RewriteNet

RewriteNet consists of an encoder that extracts decomposed style and content features, and a generator that generates a text image with the identified features. This section first describes how encoder and generator are utilized to generate an image of desired text, and explains how the modules are trained on synthetic and real datasets. Followed by it, architectural details are provided.

3.1. Inference Process

Let $\mathbf{x}_{ST}$ be the style-image with text T and style S. When a target text T' is given, our model aims to generate $\mathbf{x}_{ST'}$ whose text is switched into the target text T' from $\mathbf{x}_{ST}$ while holding its style S. To achieve the goal, RewriteNet assumes two disentangled latent features, $\mathbf{h}_S^{\text{st}}$ for the style S and $\mathbf{h}_T^{\text{ct}}$ for the content T, and conducts the content switch.

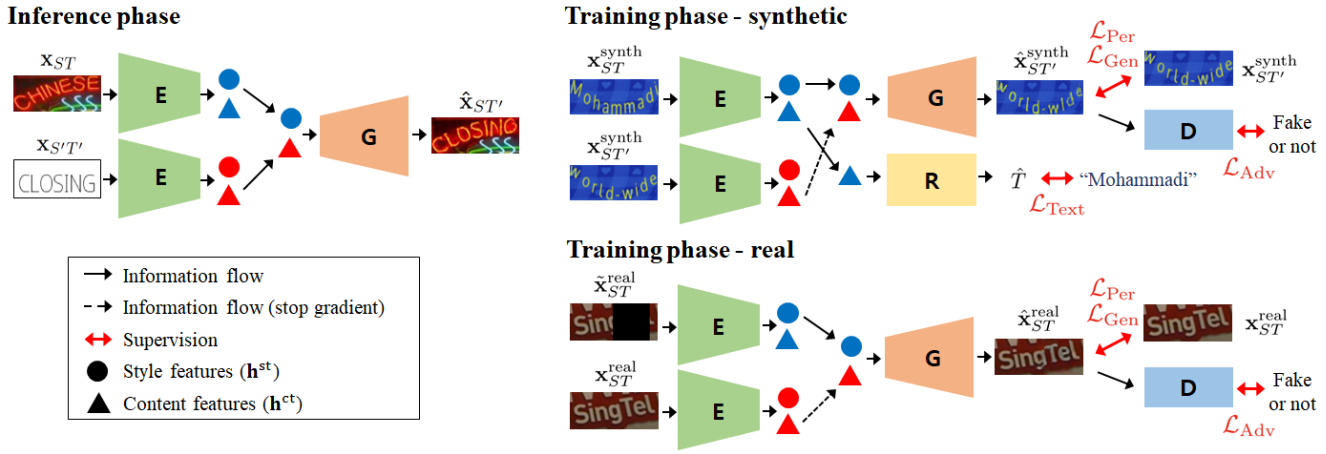


Figure 2. Overview of RewriteNet during inference and training processes. RewriteNet is composed of Style-content encoder ($\mathbf{E}$), Generator ($\mathbf{G}$), Text recognizer ($\mathbf{R}$), and Discriminator ($\mathbf{D}$). In each phase, we use the same encoder to extract style and content features from two different images. The output image is generated by combining the style feature (●) extracted from the top image (style-image) and the content feature (▲) extracted from the bottom image (content-image).

Following the encoder and generator framework, the inference model consists of the below two modules.

- **Style-content encoder** ($\mathbf{E}$: $\mathbf{x}_{ST} \rightarrow \mathbf{h}^{st}_S, \mathbf{h}^{ct}_T$) extracts latent style feature $\mathbf{h}^{st}_S$ and content feature $\mathbf{h}^{ct}_T$ from an image $\mathbf{x}_{ST}$. For better descriptions, $\mathbf{E}$ will be expressed as two terms; $\mathbf{E}^{st} : \mathbf{x}_{ST} \rightarrow \mathbf{h}^{st}_S$ and $\mathbf{E}^{ct} : \mathbf{x}_{ST} \rightarrow \mathbf{h}^{ct}_T$.
- **Generator** ($\mathbf{G}$: $\mathbf{h}^{st}_S, \mathbf{h}^{ct}_T \rightarrow \hat{\mathbf{x}}_{ST}$) generates an output image of text T under the style S.

By switching off the latent content features, the model becomes enabled to generate a text-switched image $\hat{\mathbf{x}}_{ST'}$ as follows:

$$\hat{\mathbf{x}}_{ST'} = \mathbf{G}(\mathbf{E}^{st}(\mathbf{x}_{ST}), \mathbf{E}^{ct}(\mathbf{x}_{S'T'})). \quad (1)$$

The left of Figure 2 explains the inference process in a view of the information flow. A content-image $\mathbf{x}^{synth}_{S'T'}$ is synthetically rendered with simple style S' and target text T'.

3.2. Training Process

The right of Figure 2 shows two training processes of our model. One is for paired synthetic images, and the other is for unpaired real-world images.

3.2.1 Modules Utilized in Training Process

Here, we introduce two modules only used in the training process to encourage the content-switched image generation.

- **Text recognizer** ($\mathbf{R}$: $\mathbf{h}^{ct}_T \rightarrow T$) identifies text label from the latent content feature. By learning content features $\mathbf{h}^{ct}$ to predict text label, the content feature can represent the text upon on the input image and is used as a content condition of $\mathbf{G}$. We should note that content feature $\mathbf{h}^{ct}$ is only trained with text label in the whole training process.
- **Style-content discriminator** ($\mathbf{D}$: $\hat{\mathbf{x}}_{ST'}, \mathbf{x}_{ST}, \mathbf{h}^{ct}_{T'} \rightarrow [0, 1]$) determines whether an input image $\hat{\mathbf{x}}_{ST'}$ is synthetically generated with a style reference $\mathbf{x}_{ST}$ and a content feature $\mathbf{h}^{ct}_{T'}$, where $\hat{x}_{ST'}$ is an output of $\mathbf{G}$. As a competitor of $\mathbf{G}$, its adversarial training improves generation quality.

By utilizing these modules, $\mathbf{E}$ enables to identify the latent content and the $\mathbf{G}$ enables to generate high-quality images.

3.2.2 Learning from Synthetic Data

We train the modules to decompose style and content features by using synthetic image pairs. As shown in the top right of Figure 2, synthetic image pairs share the same style but have different text contents. The content feature is learned to capture text information in an image by utilizing $\mathbf{E}^{ct}$ and $\mathbf{R}$. The encoded content feature is fed into the recognizer, and to let the recognizer predict correct labels, the encoder is trained to produce favorable content features. The style feature is learned to represent style information by allowing $\mathbf{E}^{st}$ and $\mathbf{G}$ to maintain style consistency after content switched generation.

We can obtain paired images $\{\mathbf{x}^{synth}_{ST}, \mathbf{x}^{synth}_{ST'}\}$ by feeding different texts to synthesizing engine with same rendering parameters such as background, font style, alignment, and so on. Then, a single training set becomes $\{\mathbf{x}^{synth}_{ST}, \mathbf{x}^{synth}_{ST'}, T\}$ where T is a text label. Therefore, $\mathbf{E}$, $\mathbf{G}$ and $\mathbf{R}$ can be

trained with reconstruction and recognition losses:

$$\mathcal{L}_{\text{Gen}}^{\text{synth}} = \|\mathbf{G}(\mathbf{E}^{\text{st}}(\mathbf{x}_{ST}^{\text{synth}}), \bar{\mathbf{E}}^{\text{ct}}(\mathbf{x}_{ST'}^{\text{synth}})) - \mathbf{x}_{ST'}^{\text{synth}}\|_1, \quad (2)$$

$$\mathcal{L}_{\text{Text}}^{\text{synth}} = \sum_i \text{CrossEntropy}(\mathbf{R}(\mathbf{E}^{\text{ct}}(\mathbf{x}_{ST}^{\text{synth}}))_i, T_i), \quad (3)$$

where $\bar{\mathbf{E}}^{\text{ct}}$ indicates a frozen encoder that does not get any back-propagation flow and T_i represents i-th character of the ground truth text label.

If we do not freeze $\mathbf{E}^{\text{ct}}$ at reconstruction loss, $\mathbf{E}$ and $\mathbf{G}$ will quickly fall into a local minimum by simply copying content-image. Thus, we freeze the $\mathbf{E}^{\text{ct}}$ to prevent the content feature from being affected by the reconstruction loss and train $\mathbf{E}^{\text{ct}}$ only with the recognition loss. These losses guide the model to learn the content switch, but the trained model might fail to address real-world images caused by the limitation of the synthetic styles. Here, it exists input discrepancy between training and test phases. Specifically, the input pairs share the same styles in the training phase whereas the input pairs have different styles in the inference phase. We found that training with different styles has optimization issues, and thus, RewriteNet is trained with the same style images. We will present corresponding results in a supplemental file.

3.2.3 Learning from Real-world Data

Synthetic data provides proper guidance for content switching, but it does not fully represent a style of real-world images. To reflect real-world styles, we propose a self-supervised training process for RewriteNet utilizing real-world data as shown in the right bottom of Figure 2.

In the case of real-world images, there are no paired images that have different texts with the same style. Moreover, it is expensive to obtain text labels of real-world images. Therefore, we introduce conditioned denoising autoencoder loss to allow the model to learn style and content representations for unpaired real-world images. Specifically, we cut out a region randomly selected in the width direction with length w to lose some characters [5], and then the noisy image is used as a style-image to extract the style feature from the left regions. By combining the content feature extracted from the original image, $\mathbf{G}$ fills the blank by referring to the style of the surrounding area of the blank region. The proposed self-supervised scheme will forbid the model to trivially autoencode style-image by using the corrupted image as style-image and enforce model to learn separated representations. The denoising autoencoder loss is defined as:

$$\mathcal{L}_{\text{Gen}}^{\text{real}} = \|\mathbf{G}(\mathbf{E}^{\text{st}}(\tilde{\mathbf{x}}_{ST}^{\text{real}}), \bar{\mathbf{E}}^{\text{ct}}(\mathbf{x}_{ST}^{\text{real}})) - \mathbf{x}_{ST}^{\text{real}}\|_1, \quad (4)$$

where $\tilde{\mathbf{x}}_{ST}^{\text{real}}$ indicates a noisy image corrupted from $\mathbf{x}_{ST}^{\text{real}}$. Here, we should note that the proposed self-supervised method does not require any text labels and paired images.

3.2.4 Adversarial Training

Generally, text image in the wild has high-frequency regions like complex background, diverse textures, and high contrast regions. However, pixel-wise reconstruction loss, referred to as $\mathcal{L}_{\text{Gen}}$, has a limitation to address the high-frequency and tends to capture the low-frequencies [13]. To encourage high-frequency crispness, we apply the generative adversarial network (GAN) framework to generate realistic text images [13, 18, 23, 24]. Specifically, we design the $\mathbf{D}$ to represent a fake or real probability of the input image under the given conditions of its style-image and latent content. We denote $\mathbf{D}(X|X^{\text{st}}, \mathbf{h}^{\text{ct}})$ for the probability $p(X \text{ is not fake}|X^{\text{st}}, \mathbf{h}^{\text{ct}})$, where X and X^{st} indicate the input image and the style-image respectively. The adversarial losses are calculated as follows:

$$\begin{aligned}\mathcal{L}_{\text{Adv}}^{\text{synth}} = &\log \mathbf{D}(\mathbf{x}_{ST'}^{\text{synth}}|\mathbf{x}_{ST}^{\text{synth}}, \bar{\mathbf{E}}^{\text{ct}}(\mathbf{x}_{ST'}^{\text{synth}})) \\ &+ \log\left(1 - \mathbf{D}(\hat{\mathbf{x}}_{ST'}^{\text{synth}}|\mathbf{x}_{ST}^{\text{synth}}, \bar{\mathbf{E}}^{\text{ct}}(\mathbf{x}_{ST'}^{\text{synth}}))\right), \end{aligned} \quad (5)$$

$$\begin{aligned}\mathcal{L}_{\text{Adv}}^{\text{real}} = &\log \mathbf{D}(\mathbf{x}_{ST}^{\text{real}}|\tilde{\mathbf{x}}_{ST}^{\text{real}}, \bar{\mathbf{E}}^{\text{ct}}(\mathbf{x}_{ST}^{\text{real}})) \\ &+ \log\left(1 - \mathbf{D}(\hat{\mathbf{x}}_{ST}^{\text{real}}|\tilde{\mathbf{x}}_{ST}^{\text{real}}, \bar{\mathbf{E}}^{\text{ct}}(\mathbf{x}_{ST}^{\text{real}}))\right), \end{aligned} \quad (6)$$

where $\hat{\mathbf{x}}_{ST'}^{\text{synth}}$ and $\hat{\mathbf{x}}_{ST}^{\text{real}}$ denote generated images from synthetic and real-world style-images, respectively (See Figure 2). Here, it should be noted that the latent contents used as the conditions are frozen to block back-propagation flow to the $\mathbf{E}$ from the adversarial loss.

We also employ feature matching loss that stabilizes the training of various GAN models [20, 31, 39]. Specifically, we extract intermediate feature maps of the $\mathbf{D}$ and minimize the distance between generated and target samples:

$$\mathcal{L}_{\text{Per}}^{\text{synth}} = \sum_l \frac{1}{M_l}\|\phi_l(\mathbf{x}_{ST'}^{\text{synth}}) - \phi_l(\hat{\mathbf{x}}_{ST'}^{\text{synth}})\|_1, \quad (7)$$

$$\mathcal{L}_{\text{Per}}^{\text{real}} = \sum_l \frac{1}{M_l}\|\phi_l(\mathbf{x}_{ST}^{\text{real}}) - \phi_l(\hat{\mathbf{x}}_{ST}^{\text{real}})\|_1, \quad (8)$$

where ϕ_l and M_l are the output feature map and its size of the l-th layer. For each loss, the same conditions are used to calculate the activation maps $\{\mathbf{x}_{ST}^{\text{synth}}, \bar{\mathbf{E}}^{\text{ct}}(\mathbf{x}_{ST'}^{\text{synth}})\}$ for $\mathcal{L}_{\text{Per}}^{\text{synth}}$ and $\{\tilde{\mathbf{x}}_{ST}^{\text{real}}, \bar{\mathbf{E}}^{\text{ct}}(\mathbf{x}_{ST}^{\text{real}})\}$ for $\mathcal{L}_{\text{Per}}^{\text{real}}$. Feature matching losses could facilitate $\mathbf{G}$ to match multi-scale statistics with target samples [39], thus beneficial for overall sample qualities.

3.2.5 Final Loss Term

The final losses are formalized as follows:

$$\begin{aligned}\mathcal{L} = \underset{\mathbf{E},\mathbf{G},\mathbf{R}}{\text{argmin}}\Big(&\mathcal{L}_{\text{Gen}}^{\text{synth}} + \mathcal{L}_{\text{Text}}^{\text{synth}} + \mathcal{L}_{\text{Per}}^{\text{synth}} \\ &+ \left(\mathcal{L}_{\text{Gen}}^{\text{real}} + \mathcal{L}_{\text{Per}}^{\text{real}}\right) \\ &+ \lambda \underset{\mathbf{D}}{\text{argmax}}\left(\mathcal{L}_{\text{Adv}}^{\text{synth}} + \mathcal{L}_{\text{Adv}}^{\text{real}}\right)\Big), \end{aligned} \quad (9)$$

where λ is intensity balancing the losses.

3.3. Architectural Details

Style-content Encoder The style-content encoder follows *partially shared latent space assumption* as in MUNIT [12], where an image $\mathbf{x}_{ST}$ is composed of its latent style feature $\mathbf{h}_S^{\text{st}}$ and content feature $\mathbf{h}_T^{\text{ct}}$. The network is based on a ResNet [7] similar to the feature extractor used in [3]. In addition, we apply bidirectional LSTM [9] layers upon the content features to alleviate spatial dependencies from the input image.

Text recognizer Text Recognizer estimates a sequence of characters in an image and it has an important role to distinguish contents from styles. It consists of an LSTM decoder with an attention mechanism [1] from the identified content features. Since the text labels are required to train this module, we only train the module with the synthetic dataset.

Generator Given the latent style and content features as an input, the generator outputs an image with a given style and content. The generator network is similar to the decoder used in the Unet [29] architecture. The style features in multiple $\mathbf{E}^{\text{st}}$ layers are fed into the generator using short-connections. The network design is inspired by the Pix2Pix [13] model.

Style-content discriminator The style-content discriminator determines whether an image is fake or not. The network architecture follows PatchGAN [13, 34].

4. Experiments

4.1. Datasets and Implementation Details

4.1.1 Synthetic Data for Training

Since RewriteNet requires paired synthetic datasets for supervised-learning, we generate 8M synthetic images with public synthesizing engine[1] that is based on MJSynth [14] and SynthText [6]. Specifically, we compose paired data $\{\mathbf{x}_{ST}^{\text{synth}}, \mathbf{x}_{ST'}^{\text{synth}}\}$ with same rendering parameters S such as font styles, background textures, shape for the text alignments (rotation, perspective, curve), and artificial blur noises except only for input texts (T, T'). The employed texts are the union of MJSynth and SynthText corpus and the paired synthetic dataset will be publicly available.

4.1.2 Real Data for Training and Evaluation

We combine multiple benchmark training datasets such as IIIT [25], IC13 [16], IC15 [15], and COCO [37]. The total number of training images is 59,856. Although these datasets contain ground-truth text labels, our model does not employ the text labels that are expensive in a practical scenario. For evaluation, we use the test a split of each public dataset such as IIIT [25], IC03 [21], IC13 [16], IC15 [15], SVT [38], SVTP [27] and CT80 [28] where the total number of test images is 8,536.

4.1.3 Implementation Details

We rescale the input image to 32×128 and empirically set w as 42, which is the proper length to cut out some characters and capture style information for the self-supervised training. To balance the multiple loss terms, we empirically set $\lambda = 0.1$. The model is optimized by Adam optimizer [17] with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. A cyclic learning rate [35] is applied with an initial learning rate of 1e-4 and an maximum iteration number of 300K. The batch size is 192 including 144 for synthetic data and 48 for real-world data. The total training takes 7 days using two Tesla V100s. At the inference phase, $\mathbf{x}_{S'T'}^{\text{synth}}$ is generated by $ImageDraw$ function from PIL package.

4.2. Comparison on Generation Performance

We compare our model to four models: MUNIT [12], EGSC [22], ETIW [40], and STEFANN [30]. Although MUNIT and EGSC are not specifically designed for STE task, we train the model targeted for STE task and make a comparison with our model to validate the STE performance of the representative image translation models[2]. ETIW is the exact comparison method for RewriteNet and its results are achieved from re-implementation[3]. STEFANN is designed for the character-wise correction method that requires manual text region selection, so test environments are different from other methods. We try to achieve high-quality results for STEFANN by testing multiple times with its official code[4].

In the quantitative comparison, we employ two measurements: recognition accuracy on generated images utilizing a pre-trained STR model[5], and Fréchet Inception Distance (FID) [8]. The recognition accuracy measures whether the generated images truly contain switched contents or not. The FID represents style consistency between a style-image and a generated image. Here, we would note that measurements between text switching performance and style preserving performance have a trade-off. It is because the best performance on FID is achieved when the output is the same as the input. Thus, balanced quantitative performance and visual results should be considered to compare the model

[1] https://github.com/clovaai/synthtiger

[2] We use the official codes: MUNIT(https://github.com/NVlabs/MUNIT) and EGSC(https://github.com/charliememory/EGSC-IT)

[3] https://github.com/youdao-ai/SRNet

[4] https://github.com/prasunroy/stefann

[5] https://github.com/clovaai/deep-text-recognition-benchmark

Figure 3. Visual comparisons on text-editing performance. Target texts are "09/02/2009", "EXPIRED", "BANANA", "SUNLIGHT", "system", "schedule" and "umP", respectively.

Figure 4. Generated images from RewriteNet trained with ablated training processes. "w/o SG" indicates the model is trained without stop gradient. Target texts are "exposes", "golf", "changed", "cottage" and "chocolates", respectively.

Table 1. Quantitative comparison between STE methods: "Accuracy" represents the content-switch performance (higher is better) and "FID" shows style consistency (lower is better). The bold indicates the best performance.

Models	**Accuracy** (↑)	**FID** (↓)
MUNIT [12]	80.75	65.7
EGSC [22]	0.04	43.3
ETIW [40]	31.16	**13.7**
Ours	**90.30**	16.7

performance. Quantitative performance on STEFANN is not evaluated, because it requires manual region selection for each image and it considerably takes a long time.

Table 1 presents the quantitative comparison results. The naive application of MUNIT tends not to maintain original styles, which can be confirmed in its high FID score. We observe that the naive application of EGSC would be inappropriate for STE. ETIW shows the best performance on FID, however, it achieves comparably lower accuracy. These results indicate ETIW often fails to convert the content and simply copies style-image where the examples are shown in Figure 3 (2nd and 6th rows). The proposed model achieves the best accuracy and also shows comparable performance on FID.

The visual comparisons are presented in Figure 3. MUNIT looks failed to preserve the style of style-image and ETIW tends to simply copy style-image for challenging style without content switching. STEFANN also cannot robustly edit texts when the lengths of texts are different and backgrounds are complex. In contrast, the proposed methods show promising results on multiple examples compared to other methods. More visual results are presented in Figure 1 where multi texts also can be edited by employing text region detector [2]. More visual results will be presented in supplemental file.

Table 2. Ablation study about the training processes.

Models	Accuracy (↑)	FID (↓)
Proposed	90.30	16.7
w/o $\mathbf{R}$	0.54	114.9
w/o Stop Gradient	97.22	89.9
w/o real	89.00	18.7
w/ real w/o noise	82.97	20.9

4.3. Ablation Study

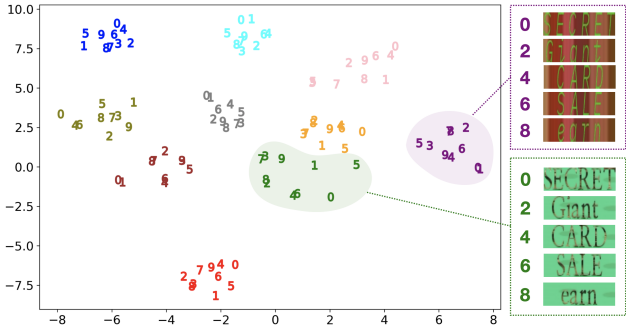
In RewriteNet, the feature decomposition is conducted by the use of a recognizer and stop gradient. Here, we describe the effectiveness of employing the recognizer and stop gradient with ablated training processes: a model without the recognizer (w/o $\mathbf{R}$) and a model without the stop gradient (w/o Stop Gradient). Table 2 and Figure 4 show the comparison results. We observe that RewriteNet cannot be trained without $\mathbf{R}$ where the performance of text switching and style preservation is dramatically degraded. "w/o Stop Gradient" achieves higher accuracy than ours, however, the performance of FID is much worse. The visual results also present the necessity of $\mathbf{R}$ and stop gradient. Specifically, "w/o Stop Gradient" simply writes the desired texts without preserving styles, which is quantitatively shown.

Moreover, we also validate the use of real data and self-supervised learning scheme with ablated training processes: a model without the training branch utilizing real-world data (w/o real) and a model feeding an original real-world image into the training branch instead of its noisy variant (w/ real w/o noise). As presented in Table 2, "w/o real" and "w/ real w/o noise" achieve lower quantitative performance than the proposed method and visual results in Figure 4 also show that the use of real data with noises generates clearer visual results when the texts are irregularly shaped and background are complex.

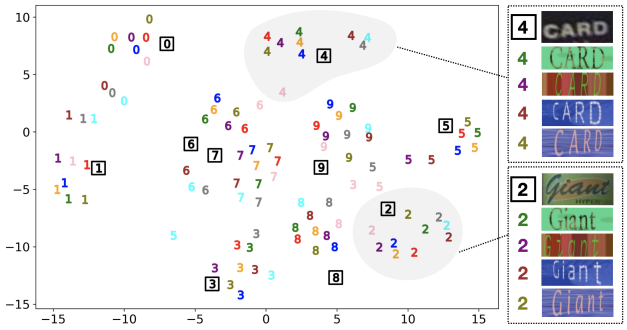
4.4. Discussions

4.4.1 Content and Style Decomposition

To validate whether our model successfully separates the content feature from the style feature, we investigate style and content features of 10×10 synthetic images (10 contents and 10 styles). Figure 5 shows the T-SNE visualizations. As can be seen, the same styles (represented with colors) are plotted closely in the style feature space and the same contents (represented with numbers) are grouped in the content feature space. In addition, we also explore the content features of real-world examples, which have the same contents as the synthetic samples, and observe that they are involved in the corresponding content clusters.



(a) T-SNE visualization of style features.



(b) T-SNE visualization of content features.

Figure 5. Visualizations of decomposed style and content features. The colored numbers indicate synthetic examples including 10 contents (numbers) with 10 styles (colors) and the boxed numbers represent real-world examples. The same styles and contents are placed closely in the corresponding feature spaces. The content features of the real-world examples are involved in the clusters holding the same text.

We also show that the style of content-image does not affect the style of the generated image to validate feature decomposition. We feed various images for content-images that have different styles with the same content and observe whether the generated results are affected. As shown in Figure 6, the generated results are quite stable to the change of content-images. Furthermore, direct text switching between real images could be conducted by switching the content features. As shown in Figure 7, "Generated A" and "Generated B" can be achieved with great visual quality when style and content images are real images. These results validate that our model well separates the content feature from the style feature of input images.

4.4.2 Robustness for Text Lengths of Contents

To show the robustness of text editing for different text lengths between desired text and style-image, we present more generation examples when the lengths of the desired texts are extremely different from the text of style-image. As can be seen in Figure 8, RewriteNet can edit different length texts robustly. Specifically, RewriteNet generates

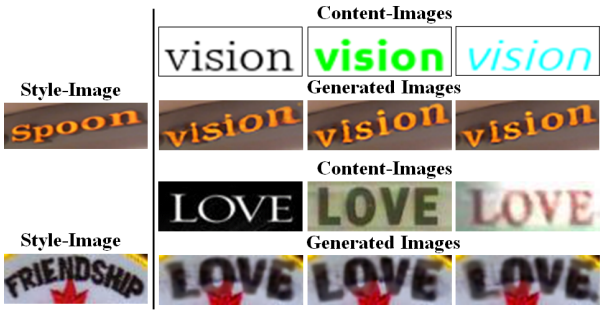


Figure 6. Generated images when given diverse content-images that have different styles with the same content

Figure 7. Text switched images when the content and style images are real scene images. "Image A" and "Image B" are the input images for generations. "Generation A" brings style and content from "Image A" and "Image B", respectively. Similarly, "Generation B" brings style and content from "Image B" and "Image A", respectively.

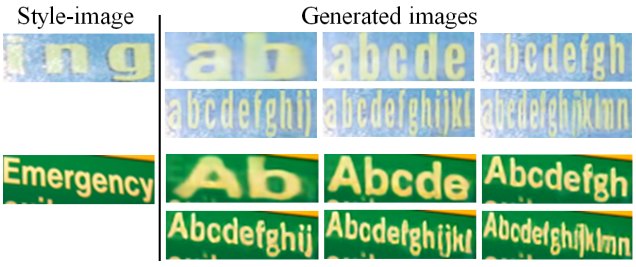


Figure 8. Generated images according to the change of lengths of the desired text.

great quality outputs (1st example) when converting the 3 characters (ing) to 14 characters (abcdefghijklmn). Interestingly, we observe that the model can properly adjust the height, width, and spacing of characters as the number of characters changed.

Table 3. STR accuracy over three benchmark test datasets depending on the training data. "Synth" indicates font-rendered data from MJSynth and SynthText. "MUNIT", "ETIW", and "Ours" represent fully generated data from unlabeled real images using MUNIT, ETIW, and RewriteNet, respectively.

Model	Train Data	IC15	CUTE80	SVTP
TRBA	Synth	78.0	76.7	79.5
TRBA	Synth+MUNIT	62.0 (↓)	57.3 (↓)	66.2 (↓)
TRBA	Synth+ETIW	64.0 (↓)	62.8 (↓)	61.1 (↓)
TRBA	Synth+Ours	**79.6** (↑)	**84.4** (↑)	**81.6** (↑)

4.4.3 Learning from Text Edited Images

It is well-known that an accurate training set leads to better performances. To evaluate the reliability of the text-edited images, we utilize the generated images for training STR models and investigate the performance gains. We train TRBA [1], a popular STR baseline, with the generated examples and rule-based synthetic images [6,14]. For the generation, four benchmark training datasets such as IIIT [25], IC13 [16], IC15 [15], and COCO [37] are employed as the style-images and the total amount of generated images is 1M. Here, we would note that STE methods do not require additional text labels for generating samples and training iterations are same for all comparisons even if the training data increases.

In Table 3, we observe other comparison methods including STE method are harmful to train STR model. These performance degradation might result from noise labels where models cannot reliably edit texts and make a mismatch between images and labels. On the other hand, the proposed RewriteNet contributes to the performance improvement all benchmarks. The results prove that RewriteNet provides more accurate and reliable examples enough to be used for training text recognition models. We will present more results according to the change of STR models [32,33] in supplemental file.

4.5. Limitations

Most of STEs including RewriteNet convert text patch-wisely, which inevitably generates unnatural boundaries around the edited patches in the entire scene images. We expect that an end-to-end scheme incorporating scene text detection and STE would relieve this problem.

5. Conclusions

This paper proposes RewriteNet which edits text in a scene image via implicit decomposition of style and content features. The novel feature decomposition methods through STR network successfully distinguish content and

style features, and their combinations are used to generate text-edited images. Thanks to the simplified pipeline, RewriteNet can utilize unlabeled real-world images with the proposed cutout strategy to reduce a gap between synthetic and real-world domains. Compared to previous STE and image translation methods, the outputs generated by RewriteNet achieve better generation quality. Further analysis demonstrates the robustness of RewriteNet on multiple content types and the reliability on the text contents in the generated images.

References

[1] Jeonghun Baek, Geewook Kim, Junyeop Lee, Sungrae Park, Dongyoon Han, Sangdoo Yun, Seong Joon Oh, and Hwalsuk Lee. What is wrong with scene text recognition model comparisons? dataset and model analysis. In *International Conference on Computer Vision (ICCV)*, 2019. 5, 8, 13, 14

[2] Youngmin Baek, Bado Lee, Dongyoon Han, Sangdoo Yun, and Hwalsuk Lee. Character region awareness for text detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9365–9374, 2019. 6, 11

[3] Zhanzhan Cheng, Fan Bai, Yunlu Xu, Gang Zheng, Shiliang Pu, and Shuigeng Zhou. Focusing attention: Towards accurate text recognition in natural images. In *ICCV*, pages 5086–5094, 2017. 5

[4] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8789–8797, 2018. 11

[5] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 4

[6] Ankush Gupta, Andrea Vedaldi, and Andrew Zisserman. Synthetic data for text localisation in natural images. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 5, 8, 14

[7] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 5

[8] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6629–6640, 2017. 5

[9] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. 5

[10] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *International conference on machine learning*, pages 1989–1998. PMLR, 2018. 11

[11] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 172–189, 2018. 2, 11

[12] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 172–189, 2018. 5, 6

[13] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017. 2, 4, 5

[14] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Reading text in the wild with convolutional neural networks. *International Journal of Computer Vision*, 116(1):1–20, 2016. 5, 8, 14

[15] 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 *ICDAR*, pages 1156–1160, 2015. 5, 8, 11, 12, 13

[16] 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 *ICDAR*, pages 1484–1493, 2013. 5, 8, 11, 13

[17] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 5

[18] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, page 1558–1566, 2016. 4

[19] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In *Advances in neural information processing systems*, pages 700–708, 2017. 2

[20] Ming-Yu Liu, Xun Huang, Arun Mallya, Tero Karras, Timo Aila, Jaakko Lehtinen, and Jan Kautz. Few-shot unsupervised image-to-image translation. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 4

[21] Simon M Lucas, Alex Panaretos, Luis Sosa, Anthony Tang, Shirley Wong, and Robert Young. Icdar 2003 robust reading competitions. In *ICDAR*, pages 682–687, 2003. 5

[22] Liqian Ma, Xu Jia, Stamatios Georgoulis, Tinne Tuytelaars, and Luc Van Gool. Exemplar guided unsupervised image-to-image translation with semantic consistency. In *International Conference on Learning Representations*, 2019. 2, 5, 6

[23] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2794–2802, 2017. 4

[24] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 4

[25] Anand Mishra, Karteek Alahari, and CV Jawahar. Scene text recognition using higher order language priors. In *BMVC*, 2012. 5, 8, 11, 13

[26] Toshiki Nakamura, Anna Zhu, Keiji Yanai, and Seiichi Uchida. Scene text eraser. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 1, pages 832–837. IEEE, 2017. 2

[27] Trung Quy Phan, Palaiahnakote Shivakumara, Shangxuan Tian, and Chew Lim Tan. Recognizing text with perspective distortion in natural scenes. In *ICCV*, pages 569–576, 2013. 5, 13

[28] Anhar Risnumawan, Palaiahankote Shivakumara, Chee Seng Chan, and Chew Lim Tan. A robust arbitrary text detection system for natural scene images. In *ESWA*, volume 41, pages 8027–8048, 2014. 5, 13

[29] 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. 5

[30] Prasun Roy, Saumik Bhattacharya, Subhankar Ghosh, and Umapada Pal. Stefann: Scene text editor using font adaptive neural network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2, 5

[31] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. Improved techniques for training gans. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 2234–2242. Curran Associates, Inc., 2016. 4

[32] Baoguang Shi, Xiang Bai, and Cong Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. In *TPAMI*, volume 39, pages 2298–2304, 2017. 8, 13, 14

[33] Baoguang Shi, Xinggang Wang, Pengyuan Lyu, Cong Yao, and Xiang Bai. Robust scene text recognition with automatic rectification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4168–4176, 2016. 8, 13, 14

[34] Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Joshua Susskind, Wenda Wang, and Russell Webb. Learning from simulated and unsupervised images through adversarial training. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2107–2116, 2017. 5

[35] Leslie N Smith. Cyclical learning rates for training neural networks. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 464–472. IEEE, 2017. 5

[36] Osman Tursun, Rui Zeng, Simon Denman, Sabesan Sivapalan, Sridha Sridharan, and Clinton Fookes. Mtrnet: A generic scene text eraser. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 39–44. IEEE, 2019. 2

[37] Andreas Veit, Tomas Matera, Lukas Neumann, Jiri Matas, and Serge Belongie. Coco-text: Dataset and benchmark for text detection and recognition in natural images. In *arXiv:1601.07140*, 2016. 5, 8, 13

[38] Kai Wang, Boris Babenko, and Serge Belongie. End-to-end scene text recognition. In *ICCV*, pages 1457–1464, 2011. 5, 11

[39] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 4

[40] Liang Wu, Chengquan Zhang, Jiaming Liu, Junyu Han, Jingtuo Liu, Errui Ding, and Xiang Bai. Editing text in the wild. In *Proceedings of the 27th ACM International Conference on Multimedia*, page 1500–1508, 2019. 2, 5, 6

[41] Wei Xiong, Jiahui Yu, Zhe Lin, Jimei Yang, Xin Lu, Connelly Barnes, and Jiebo Luo. Foreground-aware image inpainting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5840–5848, 2019. 2

[42] Qiangpeng Yang, Jun Huang, and Wei Lin. Swaptext: Image based texts transfer in scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2, 11

[43] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5505–5514, 2018. 2

[44] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017. 2, 11

[45] Jun-Yan Zhu, Richard Zhang, Deepak Pathak, Trevor Darrell, Alexei A Efros, Oliver Wang, and Eli Shechtman. Toward multimodal image-to-image translation. In *Advances in neural information processing systems*, pages 465–476, 2017. 2

A. Training and Inference Strategies

The input images for training and inference phases are slightly different in RewriteNet. Specifically, in the synthetic training phase, the inputs $\{\mathbf{x}_{ST}, \mathbf{x}_{ST'}\}$ of RewriteNet have same styles. On the other, in the inference phase, the inputs $\{\mathbf{x}_{ST}, \mathbf{x}_{S'T'}\}$ of RewriteNet have different styles. This strategy is determined empirically. In the early design choices, we found that the use of different styles in training phases could not achieve sufficient performance. The training with different styles increases training difficulties and results in unstable convergence in adversarial training. Consequently, it generates undesirable artifacts on the characters and achieves lower text-switching performance than the same styles (ours), which can be seen in Table 4. Thus, we utilize the same styled image pairs with "stop gradient" that can prevent simple auto-encoding and ensures style disentanglement.

B. Experiments

B.1. Generated Examples from Full Scene Text Images

We present more full scene generated examples by employing text detection method [2]. As shown in Figure 9, RewriteNet can successfully edit full scene images.

B.2. Comparison on Generation Performance

The recent scene text edit method is SwapText [42], however, we cannot achieve code. Thus, we simulate test environments of SwapText and we indirectly compare RewriteNet with SwapText. To compare content switching performance, we train the same recognizer model (CRNN) with real datasets [15, 16, 25, 38] and evaluate the recognition accuracy on real (original) and text switched images. In Table 5, "Real" achieves similar performance with "Real*" on SVTP and IC15, which shows that the CRNN is similarly reproduced. Then, the generated images from RewriteNet are evaluated with CRNN and the accuracy is also reported in Table 5. It shows that Ours achieves much higher accuracy than SwapText on all real datasets, which indicates that the proposed RewriteNet shows better text switching performance.

B.3. Ablation Study:Consistency Loss

Consistency loss is widely adopted in generation tasks [4, 10, 11, 44], because it can improve performance by regularizing the generator. Following the previous works, we train RewriteNet with additional consistency loss as follows:

$$\mathcal{L}_{\text{con-text}}^{\text{synth}} = \sum_i \text{CrossEntropy}(\mathbf{R}(\mathbf{E}^{\text{ct}}(\hat{\mathbf{x}}_{ST}^{\text{synth}}))_i, T_i), \quad (10)$$

Models	Accuracy (↑)	FID (↓)
Different styles	74.97	15.2
Same styles (Ours)	89.00	18.7

Table 4. Quantitative comparison between the use of same (Ours) and different style images as the training dataset: "Accuracy" represents the content-switch performance (higher is better) and "FID" shows style consistency (lower is better). Two models are trained only with synthetic data.

Datasets	SVTP	IC13	IC15
Real*	54.3	68.0	55.2
SwapText*	54.1	68.3	54.9
Real	53.0	74.2	55.3
Ours	66.0	79.7	74.6

Table 5. Text recognition accuracy on real and generated images. "Real" indicates that the input images of the recognizer are the original images. On the other hand, the input images of SwapText and Ours are generated (text switched) images. Since the codes of SwapText is not available, we bring the performances of Real and SwapText from paper, and these performances are denoted as *.

Models	Accuracy (↑)	FID (↓)
w/$\mathcal{L}_{\text{con-text}}$	94.34	16.7
Ours	90.30	16.7

Table 6. Validations of consistency loss. "Ours" indicates w/o $\mathcal{L}_{\text{con-text}}$.

where $\hat{x}_{ST}^{\text{synth}}$ denotes the generated image with the style S and content T, and T_i represents i-th character of the ground truth text label. This loss re-enforces the generated image to have desired text and it can only be applied on the synthetic data due to the requirement of the text label. The quantitative performance is reported in Table 6. It achieves a higher accuracy than the proposed, however, visual results are worse than the proposed method as can be seen in Figure 10, "w/ $\mathcal{L}_{\text{con-text}}$" sometimes erases text that is out of interest in the background (part of characters are erased in the top right region), for enhancing the recognition accuracy on the generated image. Moreover, it fails to preserve font information when the text shapes and textures are complex. From these observations, we conclude that consistency loss could not be effective to our RewriteNet and we have omitted consistency loss in the final loss term.

Figure 9. Full scene pairs of original (left) and text edited (right) images.

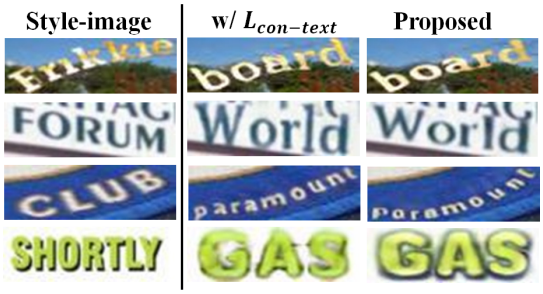


Figure 10. Visual comparisons between w/$\mathcal{L}_{\text{con-text}}$ and the proposed models.

B.4. Experiments on Feature Decomposition

B.4.1 T-SNE Visualization of Style and Content Features

We present synthetic and real images, which have been employed for visualization of T-SNE, in Figure 11 (a). We utilize a rendering tool to achieve synthetic images that have the same styles with different contents and we exploit the IC15 [15] as the real images. Figure 11 (b) and (c) show that the same styles and contents are plotted closely in the style and content feature spaces, respectively. Moreover, as shown in Figure 11 (c), we observe similar contents are located closely in the content features space where Content 6 ("SALE") and Content 7 ("sale") appear adjacent to each other.

B.4.2 Critical Components for Generation

We have validated feature decomposition between style and content in Figure 6 of the main manuscript where the generated images are quite stable to the change of content images. We investigate which component mainly affects the style of the generated image by feeding various fonts and colors of content-image. As can be seen in Figure 12, the generated images are stable according to the change of font but slightly different from each other. On the other hand, the generated images are invariant according to the change of color as shown in Figure 13. We also measure distances between the generated images and variance of the generated images with three metrics:

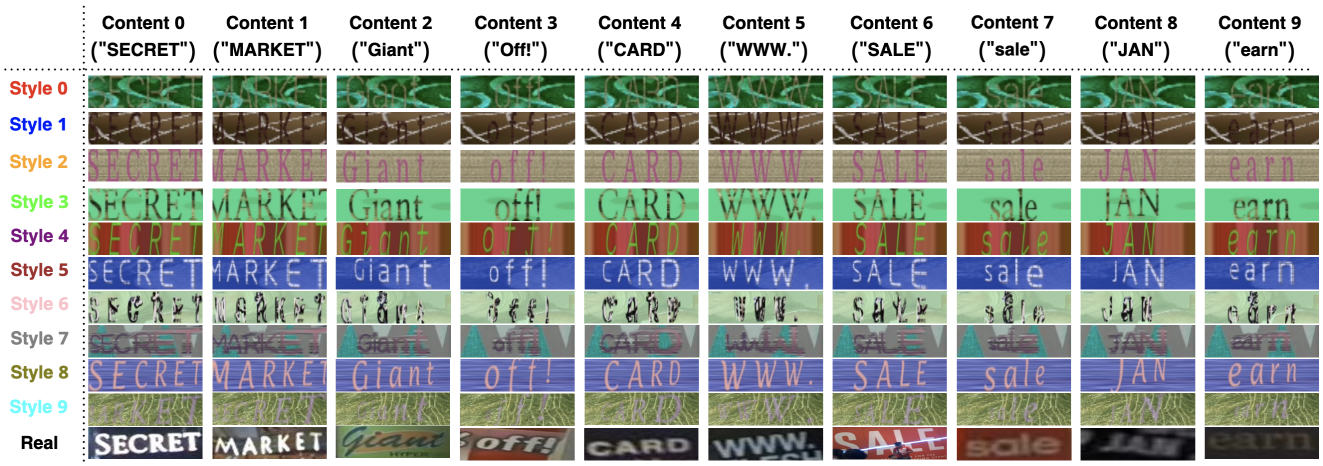
- PSNR (Peak Signal-to-Noise Ratio): pixel-wise MSE (Mean Squared Error) based distance. Since we cannot achieve the original (reference) image, we measure the distance between generated images.

- SSIM (Structural Similarity): perceptual quality-based distance. Since we cannot achieve the original (reference) image, we measure the distance between generated images.

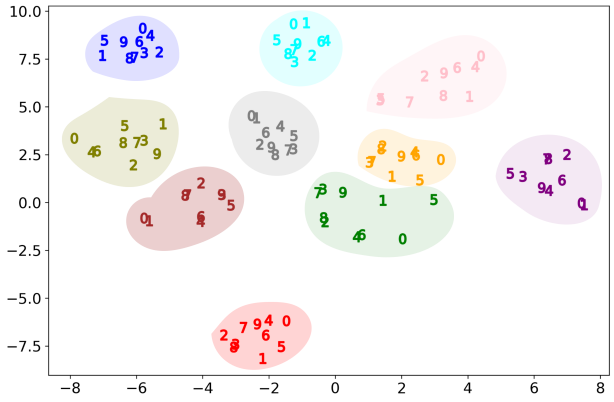
- Variance: the averaged variance of the generated images.

As can be seen in Table 7, the stability of the generated images is more affected by the change of font.

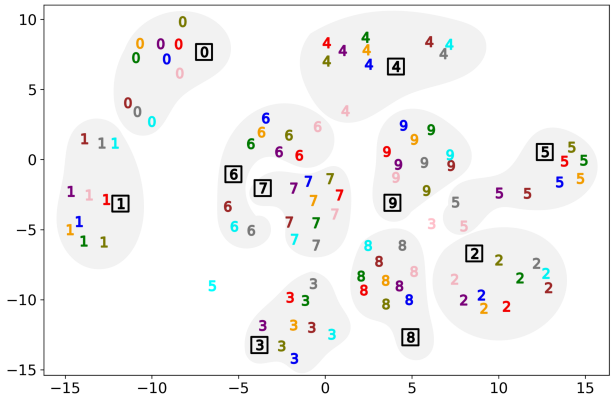
We also investigate other ablated models, which have been suspected that features are not well separated, by feeding various fonts of content-image. As shown in Figure 14, the generated images from "w/o stop gradient" significantly vary according to the change of font of content-images. Although "w/o $\mathbf{R}$" achieves stable results according to the change of content-images, its results show that content-images are not well employed.



(a) Visualizations of 10 × 10 synthesized images and 10 real images.



(b) T-SNE visualization of style feature.



(c) T-SNE visualization of content feature.

Figure 11. (a) shows input images for T-SNE. For the synthesized images, the same column and row indicate the same contents and styles, respectively. Real images of diverse styles are placed on the last row. (b) and (c) present visualizations of decomposed style and content features. The colored numbers indicate synthetic examples including 10 contents (numbers) with 10 styles (colors) and the boxed numbers represent real-world examples.

Changes	**PSNR** (↑)	**SSIM** (↑)	**Var** (↓)
Font	18.51	0.6242	7.9×10^{-3}
Color	Inf	0.9942	0.08×10^{-3}

Table 7. PSNR (dB), SSIM, and the variance between generated images. Inf denotes infinity that indicates some of the generated images are exactly identical.

B.5. Learning from Text Edited Images

B.5.1 Training and Evaluation Details

For the generation of training data, the unified real-world data (59,856 in total), combining four benchmark training datasets such as IIIT [25], IC13 [16], IC15 [15], and COCO [37], is used as the style-images. We generate 18 text images from a single style image. As a result, the total amount of generated images is about 1M (59,856 × 18).

We focus on irregular shaped real-world data, because, it is more challenging with diverse curve text alignment that could directly show the style-preservation performance. Thus, all trained models are evaluated on the three benchmarks where the total number of images is 2,744; 1,811 from IC15 [15], 645 from SVTP [27], and 288 from CT80 [28] following the evaluation protocol of scene text recognition (STR) [1].

B.5.2 Validations on Multiple STR Models

We will validate the effects of our generated data on different STR models such as CRNN [32], RARE [33] and TRBA [1]. As presented in Table 8, our generated data improves STR performances on all baselines. These results

Model	Train Data	IC15	CUTE80	SVTP
CRNN [32]	Synth	69.8	66.3	71.3
CRNN [32]	Synth+Ours	70.9	77.8	71.3
RARE [33]	Synth	75.7	73.3	75.7
RARE [33]	Synth+Ours	76.3	83.0	78.6
TRBA [1]	Synth	78.0	76.7	79.5
TRBA [1]	Synth+Ours	79.6	84.4	81.6

Table 8. Average STR accuracy of three benchmark test datasets depending on the training data. "Synth" indicates font-rendered data from MJSynth [14] and SynthText [6]. "Ours" represents fully generated data from unlabeled real images using RewriteNet.

show that RewriteNet generates reliable scene text examples, which can be well generalized to multiple STR models.

Figure 12. The generated images from RewriteNet with changing the font of content-image. The black rectangular image represents content-image and its corresponding output is listed below.

Figure 13. The generated images from RewriteNet with changing the color of content-image. The black rectangular image represents content-image and its corresponding output is listed below.

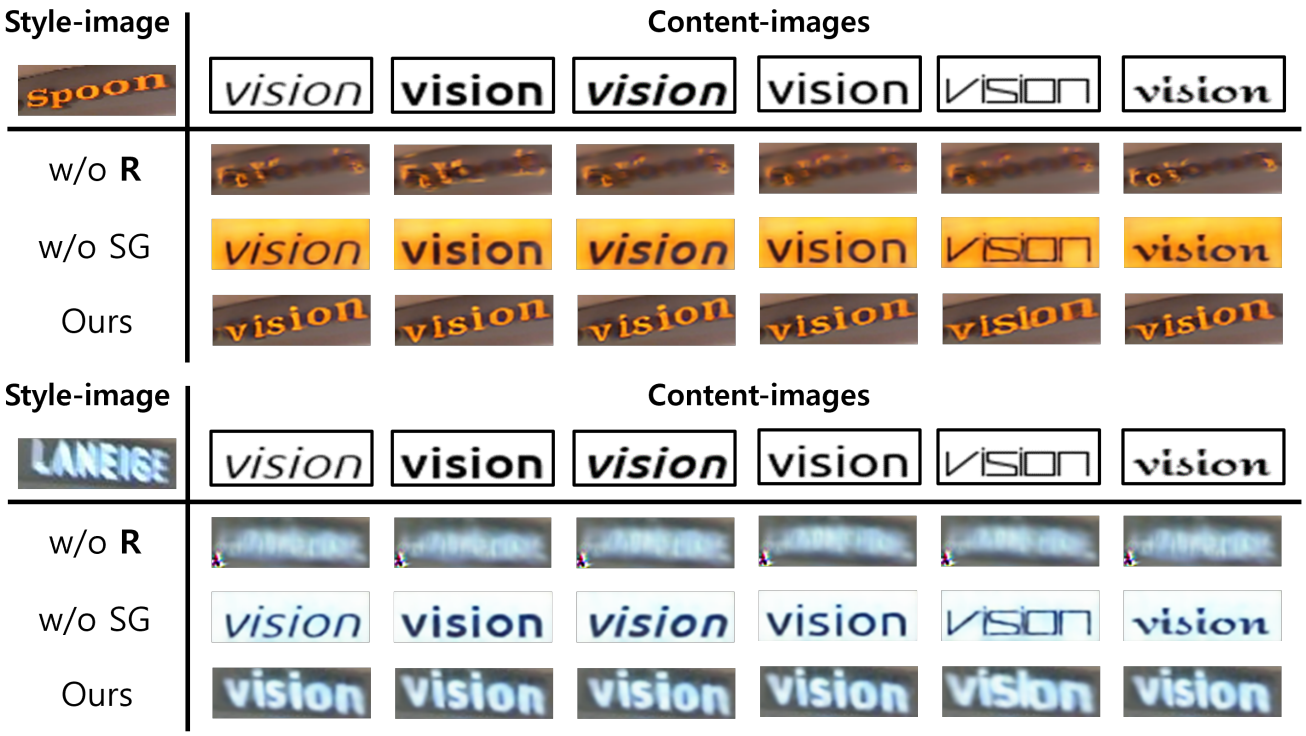


Figure 14. The generated images from "w/o **R**", "w/o stop gradient", and RewriteNet with changing the font of content-image.