Self-Supervised Text Erasing with Controllable Image Synthesis

Gangwei Jiang∗
University of Science and Technology of China
gwjiang@mail.ustc.edu.cn

Shiyao Wang
Alibaba Group
shiyao.wsy@alibaba-inc.com

Yuning Jiang
Alibaba Group
mengzhu.jyn@alibaba-inc.com

Ying Wei
City University of Hong Kong
yingwei@cityu.edu.hk

Tiezheng Ge
Alibaba Group
tiezheng.gtz@alibaba-inc.com

Defu Lian†
University of Science and Technology of China
liandefu@ustc.edu.cn

ABSTRACT
Recent efforts on text erasing have shown promising results. However, existing methods require rich yet costly label annotations to obtain robust models, which limits their use for practical applications. To this end, we study an unsupervised scenario by proposing a novel Self-supervised Text Erasing (STE) framework that jointly learns to synthesize training images with erasure ground-truth and accurately erase texts in the real world. We first design a style-aware image synthesis function to generate synthetic images with diverse styled texts based on two synthetic mechanisms. To bridge the text style gap between the synthetic and real-world data, a policy network is constructed to control the synthetic mechanisms by picking style parameters with the guidance of two specifically designed rewards. The synthetic training images with ground-truth and structure are then fed to train a coarse-to-fine erasing network. To produce better erasing outputs, a triplet erasure loss is designed to enforce the refinement stage to recover background textures. Moreover, we provide a new dataset (called PosterErase), which contains 60K high-resolution posters and is more challenging for the erasing task. The proposed method has been extensively evaluated with both PosterErase and the widely-used SCUT-Enstext dataset. Notably, on PosterErase, our method achieves 5.07 in terms of FID, with a relative improvement of 20.9% over existing supervised baselines.

CCS CONCEPTS
• Computing methodologies → Reconstruction.

KEYWORDS
text erasing, self-supervised, reinforcement learning

ACM Reference Format:
Gangwei Jiang, Shiyao Wang, Tiezheng Ge, Yuning Jiang, Ying Wei, and Defu Lian. 2022. Self-Supervised Text Erasing with Controllable Image Synthesis. In Proceedings of the 30th ACM International Conference on Multimedia (MM ’22), October 10–14, 2022, Lisboa, Portugal. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3503161.3547905

Figure 1: While the SynthText [10] is synthesized by uniform sampling from a pre-defined space, STE learns the style distribution of the real-world data by a policy network. Several erasing results on real-world data are shown on the right.

1 INTRODUCTION
Text erasing has attracted increasing interest because of its wide range of applications such as privacy protection [13], image/video editing [42], and image restoration [27]. It aims to erase text on the stroke level by filling it with a semantically plausible background [28, 36]. Most previous works [19, 27, 35, 36, 50] focus on designing a high-quality network to remove the text from natural images. For example, EnsNet [50] is the first end-to-end framework to remove text at a whole image level. MTRNet [36] utilizes an auxiliary mask to improve the text detection branch. EraseNet [19] presents a two-stage coarse-to-fine network with an additional segmentation head, which achieves state-of-the-art performance. Although they have obtained remarkable improvements, their erasing quality strongly relies on a great amount of annotated data, which requires significant economic and labor costs.

One important method to mitigate this issue is to use synthetic data. Gupta et al. [10] used an offline generation mechanism to synthesize training image pairs. As shown in Figure 1 (a), text styles are randomly drawn from a pre-defined style space. Then an image without the synthetic text will be regarded as the erasure
ground-truth of the image blending the synthetic text. However, remarkable distribution divergence between synthetic text style and existing text style in images leads to suboptimal results, as shown in Figure 1 (b). The situation becomes even worse when being adopted for high-resolution and complex poster images (see the last two rows of Figure 1 (b)).

To tackle this issue, two mainstream approaches are proposed. One is to directly generate synthetic images as realistic as real-world images based on the generative adversarial networks [26, 53]. The other is to minimize representation discrepancy between synthetic images and real-world images for style alignment [52, 54]. However, there are many diverse text styles in images for erasing, making it challenging to learn a GAN network or directly minimize representation discrepancy [9].

To address these issues, in this paper, we propose a Self-Supervised Text Erasing (STE) framework, which consists of two modules: image synthesis and text erasing. In the synthesis module, in addition to generating from a custom synthesis space as in previous work, we leverage the Maximally Stable Extremal Regions (MSER) method [23] to extract and replicate text regions from real-world images. The above customization and replication can provide sufficient variety to approximate the original text style. Then, considering the text style gap between the synthetic and real-world data, a policy network is constructed to control the synthesis function under the guidance of two well-designed rewards. The rewards are calculated in terms of the realism and difficulties of the currently selected style, encouraging the synthesis function to provide samples matching the target distribution while keeping the diversity. In the erasing module, we use a coarse-to-fine generative model to erase the text and fill missing pixels with appropriate textures. However, the current refinement network is incapable of accurately recovering detailed information when the text or background is complex. So we propose a triplet erasure loss (TEL) to solve the issue of blurry results. The TEL explicitly forces the refined results away from the coarse ones. It successfully improves the erasing results away from the ground truth. It effectively boosts the refinement network to generate more texture details and semantics than the coarse network. The synthesis module and erasing module are optimized alternately in the training process. Last but not the least, we collect 60K high-resolution poster images from the e-commerce platform to embrace more challenging scenarios for text erasing. On the whole, our STE method gains significant improvements compared to the supervised methods in both poster and scene text.

The contributions are summarized as follows:

- We propose a novel framework (STE) for text erasing, which incorporates a synthesis function and a policy network that can produce unbiased and diverse synthetic data. The controllable synthesis module ensures stable training and effectively promotes the performance of the erasing model.
- The triplet erasure loss is presented to enforce the refinement network to generate more detailed and vivid content by pushing the results away from the coarse ones. It successfully improves the erasing results of text in a complex style.
- Additionally, the first high-resolution poster text dataset for erasing is constructed (PosterErase1), containing 60K images with text detection annotations. We conduct extensive experiments on both our dataset and the public scene text dataset. Benefiting from the improvement of data synthesis and erasing model, our method significantly outperforms all other models.

2 RELATED WORK

Text Erasing: Early text erasing frameworks [16, 38] usually have two stages based on traditional text detection and image inpainting techniques while being limited to easy and single-color text. Nakamura et al. [27] initially designed a deep encoder-decoder network to erase text patch by patch, which makes a big success on account of the outstanding ability of deep learning. Zhang et al. [50] presented an end-to-end architecture to erase text across the full image based on the conditional generative adversarial network [25]. Inspired by the pix2pix model [14], Tursun et al. [36] applied text masks as auxiliary information to achieve efficient and stable training. More recently, several works [19, 35] improve the accuracy by explicitly modeling a branch that predicts the text regions and build the state-of-the-art text erasing models.

Training the above erasing models requires high-quality labeled pairs which are expensive and inefficient [19]. To address this issue, synthesizing data from non-text images [19, 35, 50] has been the most common alternative method. But the previous studies focus on the setting where training and test data are both synthetic, thus leaving the issue of dataset shift unattended.

Text Image Synthesis: Data synthesis is an economical and efficient method for data collection in deep learning, and it has been successfully applied in fields such as text detection [10], semantic segmentation [30, 31], medical analysis [6], and so on. In text image synthesis, Wang et al. [39] and Jaderberg et al. [15] first used synthetic images in text recognition tasks. Gupta et al. [10] developed an efficient engine that inserts diverse text at semantically relevant locations to synthesize text images. In [48], Zhan et al. took semantic coherence, visual attention, and adaptable text appearance into consideration achieving verisimilar text image synthesis.

A more advanced line of work resorts to the generative adversarial network (GAN). GA-DAN [49] synthesizes data by modeling the representation in geometry and appearance spaces. Wu et al. [42] and Yang et al. [45] developed an end-to-end trainable style retention network to modify text in images. These GAN-based methods obtain higher synthesis quality; however, they suffered from high computational overhead and generate images with lots of artifacts when out of the training distribution (e.g., text in shadow style).

Domain Adaptation: Domain adaptation aims to study the problem of domain shift. The majority of existing research falls into two categories. Methods like [7, 20, 21, 32, 43, 52] align the distribution of a source and target domain in the feature space. The second category aligns input images of two domains at the pixel level. Zhu et al. [53] achieved the consistency of structure and semantics by adding consistency loss on the basis of GAN models. GA-DAN [49] adds a multi-modal spatial learning model for the shifts in both geometry and appearance spaces. However, the performance of these alignment methods will decline when confronted with large variance [9]. Meanwhile, the focus on the target makes them difficult to obtain the ability outside the distribution.
3 SELF-SUPERVISED TEXT ERASING

3.1 Overview

Our goal is to learn a text erasing network $G$, which best erases texts in a target domain (original texts in the real-world image). Given the annotated images collection $\{(I, l_f)\}$ where $I$ and $l_f$ denote the real-world and corresponding annotated images, the previous supervised methods [19, 35, 50] train $G$ by solving the following optimization problem:

$$\min \theta \mathcal{L}(G_\theta(I), l_f). \tag{1}$$

where $\theta$ are the parameters of erasing model $G$, and $\mathcal{L}$ denotes the whole erase loss function generally consisting of adversarial loss $\mathcal{L}_{adv}$, reconstruction loss $\mathcal{L}_{rec}$, and so on. But in this paper, we assume that we have only available to the unlabeled image collection $\{I\}$ and study an unsupervised scenario. To train $G$, we generate the synthetic image $I_{syn}$ on $I$ using a controllable synthesis module, and leverage $\{(I_{syn}, l_f)\}$ as the training pairs (see examples in Fig. 2). We learn our model $G$ with the synthetic dataset $\{(I_{syn}, l_f)\}$, and evaluate it on a held-out labeled test set from the target domain, which is disjoint from $\{I\}$ and is never used for training.

Specifically, we present an overview of our self-supervised text erasing framework in Fig. 2. It composes of two major parts: (a) the controllable synthesis module and (b) the text erasing module. In the synthesis module, real-world image $I$ will be processed by synthesis function $F$ (Sec. 3.2), resulting in the synthesized image $I_{syn} \equiv F(I, s)$ with text instance in a specified style $s$. Then, $I_{syn}$ and its corresponding original $I$ can be regarded as a pair of input and ground-truth for training the text erasing model $G$:

$$\min \theta \mathcal{L}(G_\theta(F(I, s)), I). \tag{2}$$

Moreover, to align the source styles (synthetic text) with the target one (original text), we use a policy network $\mathcal{A}$ to select an appropriate style $s \equiv \mathcal{A}(I)$ for function $F$, which is implemented by an LSTM [12] and optimized by the environmental feedback, including text difficult $R_{diff}$ and style realistic reward $R_{real}$ (Sec. 3.3).

In the text erasing module, our model is built upon a two-stage coarse-to-fine network called EraseNet [19], where the erasing model $G$ is constructed by the coarse and refinement networks. Given the synthetic image $I_{syn}$ and a binary mask $M_{syn}$ indicating the synthesized region, the coarse network first hallucinates a rough prediction $I_r$. Then, the refinement network generates more detailed images denoted as $I_r$. And finally, the result $I_{pred}$ which is used for loss calculation is composited by $I_r$ and $I_{syn}$ conditioned on $M_{syn}$. However, the current refinement network is incapable of generating detailed texture when the coarse network recovers indistinguishable content. So we propose a novel triplet erasure loss (TEL) $\mathcal{L}_{tri}$ to ensure the refined results $I_r$ are closer to ground truth $I$ and generate more content than the $I_c$ (Sec. 3.4).

The synthesis and erasing modules are jointly trained in an end-to-end manner and thus reach better learning. Specifically, the model $G$ is optimized by gradient descent while the policy network $\mathcal{A}$ is optimized with reinforcement learning. The whole loss function of our STE method can be summarized as follows:

$$\min \theta \mathcal{L}(G_\theta(F(I, \mathcal{A}w(I))), I)$$

where $w \equiv \arg \max_w \mathbb{E}_{s \sim \mathcal{A}(I)} R(I, s) \tag{3}$

$R$ denotes the weighted sum of reward function $R_{real}$ and $R_{diff}$, and $w$ are the parameters of the policy network $\mathcal{A}$.

3.2 Style-Aware Synthesis Function

The detailed procedure of our synthesis function $F$ is shown in Fig. 3. Given an image $I$, we first utilize text detection and recognition methods [1, 5] to obtain text information like position, content, blank place, and so on. Then, we generate text with a specific style $s$ and render it in a blank place. Different from past methods [10, 48] which only use a limited customization mechanism to generate text, we propose a simple yet powerful replication mechanism to enrich the synthesis ability. Details are as follows:

**Customization mechanism:** It was proposed in [10], where the text style is decomposed into multiple separated units. According to the influence on the text style, the units can be roughly divided into the following three categories:

- Appearance unit: color, font, size, Gaussian blurring, alpha blending, and Poisson blending.
3.3 Controllable Synthesis Module

Through uniformly sampling in the large space of synthesis function $F$, we can get an offline synthetic dataset SynthRC. But the huge variance of it makes the naive training or previous synthetic-to-realistic methods difficult to work with. In this regard, we design a controllable synthesis module to generate realistic and harder training data in an online manner, capturing the variance in the target domain.

3.3.1 Search Space. The synthesis function $F$ provides various styles, all of which constitute the search space of the policy network. Let $s = \{e_1, e_2, ..., e_N\}$ denote a sample in the space where the $e_i$ denote the $i$th style element and $N$ is the number of elements. When each element is assigned a value, the synthesis function will generate a unique text image (see an example in Fig. 3). In particular, we have 20 elements, including MSER, font, color, etc., and each has a distinct selection range. Details can be found in the appendix. Consequently, multiple elements are composited combinatorially, leading to an exponential explosion in the number of the search space with almost $10^5$ candidates. Furthermore, our goal is to generate styles for each image, resulting in an unacceptable search space.

3.3.2 Style Optimization via REINFORCE. The goal of the policy network is to find a suitable synthetic style for each image in the large search space. It is a typical discrete optimization problem since the element selection in the synthesis function is not differentiable. So we formulate the selection as a reinforcement learning problem and apply the REINFORCE algorithm [41]. Concretely, the policy network is implemented as an LSTM [12]. At each synthesis process, the policy network will observe the input image $I$ as the state and predict action corresponding to a discrete parameter for each element (see the policy network part in Fig. 2). Notably, we construct our policy network as an image-aware agent, which is different from automatic data augmentation using a dataset-aware agent [3, 4]. And the objective of the policy network is to maximize the reward function as Eq. 3, and the optimization equation is as follows:

$$\nabla_w \mathbb{E}_{s \sim A} R(I, s) \approx \frac{1}{M \times N} \sum_{m=1}^{M} \sum_{n=1}^{N} R_m \nabla_w \log p^n_m. \quad (4)$$

where $M$ is the updating batch size, $N$ is the number of element and $p^n_m$ represents the probability of the choice of the $n$th element in the style of $m$-th image. $R$ and $R_m$ denotes the reward function and the reward value of the $m$-th image. The training of the policy network is inspired by [3, 51, 54], and the detailed parameter updates can be found in the appendix. Furthermore, to accommodate the hierarchical relationship in the search space (e.g., when the border style is not selected, the element controlling the border width does not work), the policy network is actually optimized using a masking mechanism as follows:

$$\nabla_w \mathbb{E}_{s \sim A} R(I, s) = \frac{1}{M \times N} \sum_{m=1}^{M} \sum_{n=1}^{N} H^n_m R_m \nabla_w \log p^n_m. \quad (5)$$

where $H^n_m$ denotes the hierarchical relationship value which is set to 1 if the element works and 0 otherwise. Details can be found in the appendix.

3.3.3 Reward Setting. To facilitate the learning of erasing module, the generated style $s$ is expected to meet the following requirements: (a) $s$ should be realistic and match with the target distribution; (b) $s$ should be challenging for the current model, providing informative gradient; and (c) $s$ should not be extremely difficult, e.g., the style that generates irregular samples is not recommended. Based on these guidelines, we designed the following rewards:

**Style realistic reward:** To capture the target distribution, we implement a text discriminator $D_{\text{text}}$ to guide the data synthesis (see
Fig. 2). Specifically, $D_{text}$ is built to predict the domain of the text region in the synthetic image $I_{syn}$. This module takes the feature map $g(I_{syn})$ of the generator $G$ as input and is compatible with any generative model plugged in.

The key to the style of realistic text design is to configure the gap between the synthetic text on the training set and the original text on the validation set collected from the target distribution. So we apply an adversarial mechanism here. In the training phase, the text discriminator $D_{text}$ aims to distinguish between the original and synthetic texts, predicting the original text as 1 and the synthetic one as 0. On the contrary, in the reward calculation phase, the closer the prediction of the synthetic text area is to 1, the greater the reward. Then the reward is formulated as (also check $R_{real}$ branch in Fig. 2):

$$R_{real}(l, s) = -L_{dice}(D_{text}(g(I_{syn})), M_{syn})$$

where $L_{dice}$ is the dice loss [24] for mask image and $M_{syn}$ is the mask of synthetic text region. The realistic reward discourages style shifts and ensures that the synthetic data is within the same distribution as the target data.

Text difficult reward: Inspired by the success of adversarial training in distributionally robust optimization [33, 37], we consider increasing the training loss of the erasing model with hard and diverse synthetic samples. Training by the cases with larger losses guarantees robust performance against data distributions. Meanwhile, to reduce the risk of training collapse, we refer to the approach in [18, 22], and finally propose a text difficult reward as follows (also check $R_{diff}$ branch in Fig. 2):

$$R_{diff}(l, s) = -|1 - \exp [L - \alpha L_{mean}]]$$

where $L$ is the L1-distance between output image $I_{pred} = G(F(l, s))$ and original image $l$, and $L_{mean}$ is the exponential average of historical distance. And $\alpha > 1$ controls the difficult level of generated samples, constraining $L$ stays in a certain range, i.e., $\alpha L_{mean}$.

Finally, the REINFORCE is employed to update the policy network using the weighted sum of the above rewards, while $R_{real}, R_{diff}$ are normalized. In practice, we update the policy network $A$ for every certain number of model iterations, which makes the time overhead caused by the guided synthesis module negligible.

$$R = \alpha_1 R_{real} + \alpha_2 R_{diff}$$

### 3.4 Erasing Module with Triplet Erasure Loss

In this section, we will discuss how our erasing module and the newly designed loss work. The model in STE aims to explore the in-variance between target and source, to build the erasing ability of real-world data. However, as the labels in the target domain are not given, they may still suffer from erasure blurring, even when adopting the coarse-to-fine generator (see in Fig. 2). The reason likely lies in that the current losses do not differentiate outputs of the two stages, causing the model to learn similar representations, which leads to tiny changes or fuzzy textures in the outcomes.

Inspired by the recent progress of contrastive learning, we propose our novel triplet erasure loss (TEL) to explicitly enforce the refinement network to learn different representations from the coarse network. It can be formulated as:

$$\mathcal{L}_{te} = \frac{||l_r - l||_1}{||l_r - l||_1 + ||l_r - detach(l_c)||_1}$$

where $l, l_r, l_c$ are the label image, coarse output, and refined output, respectively. Specifically, we stop gradient into $l_c$ by detach operation and $y$ is the scaling factor that emphasizes the impact of hard samples. Intuitively, TEL ensures the refined results are closer to ground truth and generates better images than the coarse results.

Minimizing TEL addresses the blurring erasure problem from two perspectives. The first is that minimizing the term $||l_r - l||_1$ directly constrains the refined image standing close to its corresponding ground truth. The second is that it guides the refinement network to take a further leap from $l_c$ and generates more detailed results by maximizing the distance between them. Consequently, the loss adds a pushing force from the “negative samples” $l_c$, and the ground truth image provides an optimal direction to pull $l_r$ close to it, making the refinement network easier to converge. Notably, these two objectives mutually improve via optimization.

The training process for the erasing module has been described in Sec 3.1. Unlike the previously supervised learning [19, 50], our training data $(I_{syn}, l)$ contains labeled synthetic text and unlabeled target text in the same image. So when calculating losses, our model is expected to focus only on the synthetic text region. Thus, we combine the output image $l_r$ and input image $I_{syn}$ conditioned on the synthetic mask $M_{syn}$ to create our final prediction $I_{pred}$:

$$I_{pred} = M_{syn} \times I_{syn} + (1 - M_{syn}) \times l_r$$

Specifically, $M_{syn}$ is generated by the synthesis function without any prediction.

Finally, we adopt the adversarial loss $L_{adv}$, reconstruction loss $L_{rec}$, perceptual loss $L_{per}$, style loss $L_{sty}$, mask refined loss $L_m$, and triplet erasure loss $L_{te}$ as the learning objectives. The whole loss function for erasing model can be summarized as follows:

$$\mathcal{L}(I_{pred}, l) = \lambda_1 L_{adv} + \lambda_2 L_{rec} + \lambda_3 L_{per} + \lambda_4 L_{sty} + \lambda_5 L_m + \lambda_6 L_{te}$$

where $\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5,$ and $\lambda_6$ are trade-off hyper-parameters.

### 4 EXPERIMENT

In this section, we conduct experiments on PosterErase and SCUT-Enstext [19] datasets to investigate the erasing quality, synthesis ability, and robustness of our self-supervised text erasing method. Unless otherwise specified, the reported results below are the performance tested on the PosterErase dataset.

#### 4.1 Dataset

**SCUT-Enstext:** It is a widely used benchmark for scene text erasing collected by [19]. It contains 2,748 training images and 813 test images with their annotations. Since our method is a self-supervised framework, it can benefit from increased amounts of unlabeled text images. So we adopt 10,166 scene text images without annotations from ICDAR2019-Art dataset [2] as auxiliary information. All the images are distinct and resized to 512 x 512.

**PosterErase:** The dataset is collected on the e-commerce platform and consists of posters mainly with Chinese text. The dataset contains 60,000 training images and 400 test images. Each image is released with its text information, including the bounding box and
text content. We also provide the annotated images for the test set, carefully processed by human experts using Adobe Photoshop (PS). Through the test dataset, we can quantitatively evaluate the quality of the model. Each image is in the size of $750 \times 513$ and is resized to $768 \times 512$ as the model input.

4.2 Evaluation Metrics & Implementation Details

Three commonly used image generation metrics are used to evaluate the quality of the method: PSNR, SSIM, and FID. PSNR describes the error between pixels, SSIM [40] evaluates the structural similarity of the two images, and FID [11] compares the quality of the generated pictures from the feature level. The larger PSNR, SSIM, and the smaller FID symbolize the higher the quality of model generation. In addition, we also provide the visualization results (Fig. 4) to show the model’s effect directly.

We use EraseNet [19] as our erasing model backbone and the policy network is implemented by a two-layer LSTM. The experimental results are obtained from the same initial training point. And more details can be found in the appendix.

4.3 Comparison with the state-of-the-Art

In this subsection, we show the performance of the related methods in Table 1 and Figure 4. The baseline methods can be divided into two groups: supervised text erasing models and synthetic-to-real adaptation methods.

- **Supervised text erasing models**: the first includes the various supervised/weak-supervised erasing models, including Pix2pix [14], EnsNet [50], MtrNet++ [35], EraseNet [19] and SceneTextErase [47]. They are trained on the annotated data and then tested. For the PosterErase dataset which only contains 150 training pairs, we take the well-trained model on SCUT-Enstext as initialization and achieve better results.

- **Synthetic-to-real adaptation models**: the second type focuses on the utilization of synthetic data. SynthText is the offline synthetic dataset proposed by [10], while DANN [54], PCD [46], AFN [43], and AA [51] are a series of methods dealing with dataset shift. Specifically, DANN and PCD capture the consistency of features between domains, AFN adopts feature-norm across domains, and AA applies adversarial data augmentation. These methods are trained on the synthetic dataset SynthRC and then tested on real-world images.

The hyper-parameters of each baseline are carefully designed to ensure the best possible results.

We compare Self-supervised Text Erasing to the above baselines on both PosterErase and SCUT-Enstext datasets. Specifically, SynthRC is the offline synthesis method proposed in this paper which is based on replication and customization mechanism. The experimental results are reported in Table 1. The key finding is that the STE algorithm achieved excellent erasing performance without annotated data by effectively solving the inconsistency between the synthetic samples and the target data while providing diversity. Other findings include the following:

**Quantitative evaluation**: Firstly, our algorithm outperforms the state-of-the-art supervised methods in text erasing, with improvements on FID of 20.9% and 10.3% in PosterErase and SCUT-Enstext. This is because the supervised models are learned by limited labeled data, while STE provides the method for introducing richer
Table 1: Comparison with baselines in PosterErase and SCUT-Enstext dataset. Higher PSNR, SSIM, and Lower FID is better.

|                  | PosterErase | SCUT-Enstext [19] |
|------------------|-------------|-------------------|
|                  | SSIM(↑)     | PSNR(↑)           | FID(↓) | SSIM(↑) | PSNR(↑) | FID(↓) |
| Text Erase       |             |                   |        |         |         |        |
| Pix2pix [14]     | 0.9160 26.828 7.672 | 0.5343 23.865 17.05 |
| EnsNet [50]      | 0.9399 30.196 7.409 | 0.8245 30.785 6.548 |
| MtrNet++ [35]    | 0.8326 26.921 10.11 | 0.8754 30.982 7.379 |
| EraseNet [19]    | 0.9389 34.034 6.411 | 0.8844 32.092 5.567 |
| SceneTextErase [47] |            |                   |        |         |         |        |
| Synthetic to Realistic |            |                   |        |         |         |        |
| SynthText [10]   | 0.9363 32.042 6.599 | 0.8796 31.839 6.069 |
| DANN [7]         | 0.9433 35.728 5.625 | 0.8837 32.939 5.395 |
| PCD [46]         | 0.9457 35.601 5.622 | 0.8823 32.751 5.542 |
| AFN [43]         | 0.9463 36.098 5.269 | 0.8827 32.650 5.675 |
| AA [51]          | 0.9353 33.477 6.219 | 0.8812 32.669 5.928 |
| Ours             |             |                   |        |         |         |        |
| SynthRC          | 0.9418 34.707 5.945 | 0.8829 32.765 5.480 |
| STE              | 0.9548 37.249 5.070 | 0.8867 33.198 4.990 |
| STE+Finetune     | **0.9648 39.914 4.093** | **0.8915 34.139 4.318** |

training materials. Second, when compared with the algorithms in the synthetic-to-realistic topic, our method still achieves significant improvements on FID of 3.8% and 7.5% in two datasets. By regulating the synthesis process, our method could match the variance in target from the large space of synthesis function and provide generation ability to out-of-distribution images. Thus, the erasing model can be trained stably and fast, and perform well in the target scene. Finally, we tried to evaluate our STE in a semi-supervised setup by unsupervised pretraining followed by finetuning using a small number of labeled samples. The result in the last row of Table 1 demonstrates that our self-supervised model with massive unlabeled data provides an effective prior for model initialization and helps to yield better performance.

Quantitative comparison: We also present several erasing examples on both datasets in Fig. 4. The erasing results of EraseNet [19] and DANN [7] are compared to our outcomes as they have the best erasing performance in the two types of baselines. It is evident that STE generates the most visually appealing results with the fewest noticeable artifacts. More findings are as follows. First, the model from supervised training (i.e., EraseNet in the figure) performs poorly in detecting the text area and completing the background. This is primarily due to the limited scale of training data, which makes the model difficult in generalizing to new cases. Second, the DANN lacks the capacity to process special style texts (e.g., the shadow on the case in the second row of the first column and the border on the cases in the first to second rows of the second column have not been erased), but our method improves the erasing results of these images by altering the synthetic data distribution.

4.4 Ablation Study
To validate the effectiveness of each component, we conduct comprehensive ablation studies in this section. The majority of the experiments were performed on the PosterErase.

Triplet erasure loss: We employ triplet erasure loss $L_{te}$ to help the refinement network in producing clear texture. To demonstrate its superiority, we compare it with the baseline without using $L_{te}$.

Table 2: Ablation study on triplet erasure loss (TEL).

|                  | PosterErase | SCUT-Enstext |
|------------------|-------------|--------------|
| w/o $L_{te}$     |             |              |
| psnr | fid | psnr | fid |
| w/o $L_{te}$ | 36.796 | 5.277 | 32.947 | 5.264 |
| with $L_{te}$   | 37.249 | 5.070 | 33.198 | 4.990 |

Effectiveness of style optimization components: Table 3 shows the impact of each component in the optimization process of the controllable synthesis module. We study three variants: 1) using only realistic reward $R_{real}$; 2) using both two rewards $R_{real}$ and $R_{diff}$; 3) using the image as the environmental state in REINFORCE. Experiments show that the complete optimization design has gained 2.09 improvements on PSNR. We also find that $R_{real}$ helps in finding the style in target distribution, whereas $R_{diff}$ is more likely to yield complex samples, and the final reward function effectively combines the benefits of both. Here $R_{real}$ takes the greatest role in guiding the synthesis of training data. On the other hand, considering the
image as a state enriches the ability of the synthesis module to model text styles, capturing the personalized demands, and hence improving the synthesis performance.

Table 3: The influence of the component in the optimization of the controllable synthesis module. Image-aware indicates that the optimization takes the image as the state.

| R_{real} | R_{diff} | image-aware | PSNR(T) | FID(J) |
|----------|----------|-------------|--------|--------|
| ✓        |          |             | 34.71  | 5.946  |
| ✓        | ✓        |             | 36.07 (11.36) | 5.612 (0.334) |
| ✓        | ✓        | ✓           | 36.29 (11.58) | 5.483 (0.463) |
| ✓        | ✓        | ✓           | 36.80 (12.09) | 5.277 (0.669) |

Text style output by controllable synthesis module: A major contribution of our work is to guide the synthesis function to automatically produce realistic and harder training pairs. Therefore, we visualized the synthesis probabilities of each element in the output text style. The probability value is obtained by counting the element selection of 2,000 images under the guidance of the policy network. The results are shown in Fig. 6. The parameter choices of each element are represented by "choice0-11", where the detailed information is listed in the appendix. Generally, the higher the number, the deeper the degree of operation. For example, "Alpha0-3" represents 0-30% of the background involved during alpha blending. We can see that there is a significant difference in the elements of gaussian, alpha, curve, and perspective between the two datasets, which is congruent with our common sense of natural and poster scenarios. The results reveal that our controllable synthesis module is effective in resolving the dataset shift issue.

Robustness to different style spaces: We also check the performance of the STE in different spaces of the synthesis function $F$ to confirm its effectiveness. The results are summarized in Fig. 7. We construct four pre-defined spaces and detailed information is available in the appendix. As the style space rises, the variance in the source domain increases and the performance of baselines improves and then declines. Instead, our STE continually maintains efficiency and significantly outperforms them. Here, the drop of SynthRC on space2, 3 is related to the introduction of the "Perspective" element which rarely occurs in the poster scene.

Analysis on the synthesis function: We demonstrate the effectiveness of each component in the synthesis function (also see Sec. 3.2), and the results are reported in Table 4. By adding the replication mechanism, and the customization mechanism in the synthesis function, the performance of the trained model can raise from 27.518 to 34.707. Specifically, they are all trained without the guides of the policy network. Such a result shows that with the various text styles in the target task that can be effectively included in the synthesis function, higher performance of the model training can be achieved. Notably, replication works better in natural scenes. This is because the text changes of natural scenes are concentrated on appearance and geometry, which can be well copied by MSER.

5 CONCLUSION

In this paper, we propose a novel framework, named Self-supervised Text Erasing (STE), to learn the generation of training image pairs in an unsupervised fashion for the text erasing task. Specifically, STE method is composed of two newly developed modules: text synthesis and text erasing. The synthesis module is responsible for fast generating training samples as well as learning a policy network for steering the synthesis process by selecting more realistic and harder training data. The text erasing module employs a so-called triplet erasure loss to better recover background textures. The two modules are alternately updated in the training process. We collect 60K high-resolution poster images from the e-commerce platform to embrace more challenging scenarios for text erasing. Extensive experiments clearly show that our unsupervised STE method outperforms existing supervised baselines, which demonstrates the effectiveness of the text synthesis and erasing strategy.

ACKNOWLEDGMENTS

The work was supported by Alibaba Group through Alibaba Innovative Research Program. Defu Lian is supported by grants from the National Natural Science Foundation of China (No. 61976198 and 62022077).
We use EraseNet [19] with its default setting as our erasing model. \( \alpha \) weights are set to 0.00001, 0.00005 in the first state) to measure the quality of policy network under policy. The text area in artistic style does not meet the definition of the stable region in the MSER method. As a result, there are a number of problematic cases especially in the poster scene, i.e., the inability to extract the text pixels. We show some examples in Fig. 9.

### C IMPLEMENTATION DETAILS OF POLICY NETWORK

Due to the discrete nature of the search space, we employ a policy network to optimize the output style of the synthesis module, using the REINFORCE algorithm [41]. The policy network is implemented as a two-layer LSTM [12] in the experiment, with the hidden layer size set to 100. The probability of the action of each element will be output bypassing the result of the LSTM network to the softmax layer, and then the predicted action will be turned into a vector by the embedding layer as the input of the next layer. Specifically, the output of the first layer is from image \( I \), which is considered as a state. The two processes are repeated until all of the elements have been selected (also see Fig. 2 in the paper). Mathematically, the style \( s = \{e_1, e_2, ..., e_N\} \) is the recapitulation of the selected actions and \( e_i \) denotes the action for each time step (element) while \( N \) is the number of elements. The whole episode is formulated as \( \{(x_1, e_1, 0), (x_2, e_2, 0), ..., (x_N, e_N, R(I, s))\} \) where the triples represent state, action, and reward, respectively. Specifically, \( x_1 \) is the input image \( I \) and \( x_i (i \neq 1) \) is the embedding of latest action \( e_{i-1} \). Hence, we can use the start value \( v_1 \) (i.e., the accumulated reward of the first state) to measure the quality of policy network \( \mathcal{A} \) and the objective is as follows:

\[
J(w) = V_{\pi_w}(x_1) = \mathbb{E}_{\pi_w} [v_1] = \mathbb{E}_{s \sim \mathcal{A}_w(I)} R(I, s),
\]

where \( \mathcal{A}_w \) denotes the sequence output of the LSTM network and \( \pi_w \) denotes the output of the LSTM network at each step.

Then according to the policy gradient theorem [34], the optimization of the policy network can be formalized as:

\[
\nabla J(w) = \sum_x d_{\pi}(x) \sum_e q_{\pi}(x, e) \nabla \pi_w (e|x) = \mathbb{E}_{\pi} \left[ \sum_e \mathbb{Q}_{\pi}(x_n, e) \nabla_w \pi_w (e|x_n) \right].
\]

In this way, the replication mechanism realized by the EraseNet method is able to imitate the text style in the dataset to synthesize training samples. However, this method has its limitations, since there are many types of artistic styles, such as border and shadow. The text area in artistic style does not meet the definition of the stable region in the MSER method. As a result, there are a number of problematic cases especially in the poster scene, i.e., the inability to extract the text pixels. We show some examples in Fig. 9.

Figure 9: Examples that MSER cannot extract efficiently. The first/second row is the text in border/shadow style.

### B DETAILS OF MAXIMALLY STABLE EXTREMAL REGIONS

Maximally Stable Extremal Regions (MSER) [23] is a traditional method for detecting spots in images, and it has been widely used in text detection [8, 44]. It is capable of locating a pixel-level text region in a given area. Therefore, we choose MSER to extract the text style in the image, and the procedure is summarized as Fig. 8. First, we use the MSER on the image area to get the stable regions and then adopt a set of rules including non-maximum suppression [29] to improve the success rate of the extraction.

1. Input Region
2. MSER Output
3. Filter by Rules
4. Extracted Mask
5. Extracted Text

Figure 8: The main procedure of Maximally Stable Extremal Regions is shown through two examples, where each box in the second or third step indicates a stable area found by the algorithm.
Table 5: The search space for policy network. The numerical range in the table is a discrete range and includes both sides, e.g., 0-1 represents 0, 1 is both optional.

| No | Element Name | Description | Range | Content | Space0 | Space1 | Space2 | Space3 |
|----|--------------|-------------|-------|---------|--------|--------|--------|--------|
| 0  | MSER         | Text style generation methods | 0-1   | no/yes  | 0-1    | 0-1    | 0-1    | 0-1    |
| 1  | Font         | The font of text              | 0-11  | 12 fonts| 0-12   | 0-12   | 0-12   | 0-12   |
| 2  | Color        | The color of text             | 0-2   | 3 cluster colors | 0-2 | 0-2 | 0-2 | 0-2 |
| 3  | Size         | Change font size              | 0-4   | -(8-15)/-(1-7)/0/(1-7)/+(8-15) | 2 | 1-3 | 1-3 | 0-4 |
| 4  | Gaussian     | Gaussian blurring for text    | 0-3   | kernel size:0/1/3/5 | 0-1 | 0-2 | 0-2 | 0-3 |
| 5  | Perspective  | Perspective transformation    | 0-4   | 0/4 angels | 0 | 0 | 0-2 | 0-4 |
| 6  | Iter         | Iter text                    | 0-1   | no/yes  | 0-1    | 0-1    | 0-1    | 0-1    |
| 7  | Curv         | Curved text                  | 0-1   | yes/no  | 0-1    | 0-1    | 0-1    | 0-1    |
| 8  | Alpha        | Alpha blending               | 0-3   | 0/0.1/0.2/0.3 | 0-1 | 0-2 | 0-2 | 0-3 |
| 9  | Pois         | Poisson blending             | 0-1   | no/yes  | 0-1    | 0-1    | 0-1    | 0-1    |
| 10 | Art          | Artistic structure           | 0-1   | no/yes  | 0-1    | 0-1    | 0-1    | 0-1    |
| 11 | Border       | Border structure             | 0-1   | yes/no  | 0-1    | 0-1    | 0-1    | 0-1    |
| 12 | Border Color | The color of border          | 0-2   | 3 colors | 0 | 0-2 | 0-2 | 0-2 |
| 13 | Border Width | The width of border          | 0-5   | 2/5/7/9/11/14 | 1 | 0-3 | 0-4 | 1-5 |
| 14 | Shadow       | Shadow structure             | 0-1   | no/yes  | 0-1    | 0-1    | 0-1    | 0-1    |
| 15 | Shadow Color | The color of shadow          | 0-2   | 3 colors | 0 | 0-2 | 0-2 | 0-2 |
| 16 | Shadow Angle | The angle of shadow          | 0-7   | 8 angels | 1 | 0-7 | 0-7 | 0-7 |
| 17 | Shadow Distance | The distance of shadow     | 0-6   | 1/3/5/7/9/12/15 | 1 | 0-3 | 0-4 | 1-6 |
| 18 | Shadow Alpha | Alpha blending for shadow    | 0-4   | 0/0.1/0.3/0.5/0.7 | 0 | 0-4 | 0-4 | 0-4 |
| 19 | Shadow Blurring | Gaussian blurring for shadow | 0-4 | kernel size:0/6/9/12/15 | 0 | 0-4 | 0-4 | 0-4 |

with the collection of episodes, we can obtain the update under the assumption of REINFORCE (i.e., return as an unbiased sample of the policy value, \( q_\pi(x, e) = R(I, s) \)):

\[
\nabla J(w) = E_\pi \left[ \sum_e q_\pi(x_n, e) \pi_w(e|x_n) \nabla_w \log \pi_w(e|x_n) \right] \\
= E_\pi \left[ q_\pi(x_n, e_n) \nabla_w \log \pi_w(e_n|x_n) \right] \\
= E_\pi \left[ R(I, s) \nabla_w \log \pi_w(e_n|x_n) \right] \\
= \frac{1}{M+N} \sum_{m=1}^{M} \sum_{n=1}^{N} \tilde{R}_m \nabla_w \log p_m^n, \tag{14}
\]

where \( s_m \) denotes the sample style of the \( m \)-th image and \( p_m^n \) represents the probability of the element selection \( e_n \) of the \( m \)-th image (i.e., \( p_m^n = \pi_w(e_n|x_n) \)). \( \tilde{R}_m \) is the reward value in the \( m \)-th episode, i.e., \( \tilde{R}_m = R(I_m, s_m) \). In addition, we normalized the rewards \( \tilde{R}_m \) as \( \tilde{R}_m \) to achieve stable training while introducing a hierarchical relationship in the reward. Hence, the final optimization of the policy network can be formalized as:

\[
w_{i+1} = w_i + \frac{1}{M+N} \sum_{m=1}^{M} \sum_{n=1}^{N} \tilde{R}_m \ast H_m^n \nabla_w \log p_m^n, \tag{15}
\]

where the detail information about \( H \) can be found in Sec. D.

D DETAILS OF SEARCH SPACE

The design of the search space is essential for such a framework. It has to cover as many text styles as possible from the real-world data. And a search space consists of two parts: the types of elements that make up the style, and the selection range for each element. Our work employs four search spaces for both PosterErase and SCUT-Enstext datasets. We list the specific information of each search space in Table 5.

In addition, there will be a hierarchical relationship between the elements in the synthesis function. For example, when you select the MSER operation to do replication, all other elements will not have an effect on the final generated style. At this point, it would be incorrect if the model was still updating parameters based on all elements. Therefore, we introduce a new variable \( H \) which will be set to 1 when the element is active to the style, and 0 otherwise. The specific hierarchical relationship is as follows:

- **MSER**: if set to 1, the \( H \) of all the other elements is set to 0.
- **Art**: if set to 0, the \( H \) for the elements 11-19 in Table 5 is set to 0.
- **Border**: if set to 0, the \( H \) for the elements 12-13 in Table 5 is set to 0.
- **Shadow**: if set to 0, the \( H \) for the elements 15-19 in Table 5 is set to 0.