REAL IN, REAL OUT: What If We Only Use Real Data for Scene Text Editing?

Xingsong Ye[1,2], **Yongkun Du**[1,2], **Jiaxin Zhang**[3], **Chong Sun**[3], **Zhineng Chen**[1,2*]
[1]Institute of Trustworthy Embodied AI, Fudan University
[2]Shanghai Key Laboratory of Multimodal Embodied AI [3]WeChat Vision, Tencent Inc.

Abstract

Aiming to modify textual content while preserving the original style, Scene Text Editing (STE) has significant practical value in many applications. STE typically requires paired images before and after editing for training. However, recent approaches rely heavily on synthetic paired data and complex disentanglement strategies. They struggle to faithfully capture the diversity and complexity of natural scenes, often producing artifacts or hallucinations. In this work, we challenge the common practice and investigate whether STE can be driven by turning each real sample into its own paired supervision. Concretely, we explore multiple ways to disrupt the original image (e.g., Crop, Shuffle, Mask) to construct self-paired training data, and build a strong baseline (**TextRIRO**) upon a conditional diffusion architecture. Our experiments show that combining character-column shuffling with image cropping (Shuffle&Crop) effectively destroys text semantics while preserving key style cues, enabling the model to learn editing behaviors directly from the real text image. With this concise yet real-oriented design, TextRIRO achieves promising performance in both accuracy and visual realism across real-world benchmarks, substantially mitigating the synthetic-to-real domain gap. Furthermore, we leverage TextRIRO to build **TextRIRO-3M**, a large-scale realistic dataset for long-text recognition by concatenating edited and original images. Experimental results show that training recognition models with TextRIRO-3M significantly improves recognition accuracy, demonstrating that TextRIRO provides highly useful training resources for downstream tasks. Overall, TextRIRO lifts STE to realistic, reliable, and widely applicable new levels. Code is available at `https://github.com/YesianRohn/TextRIRO`.

1 Introduction

Scene Text Editing (STE) aims to extract the style of the given text and replace it with a new string, while maintaining both visual realism and style consistency Wu et al. [2019]. STE is typically formulated as a generative task, and its applications grow with the advancement of generative models. It addresses practical needs such as replacing text in images for artistic creation and also facilitates various downstream scene text tasks. Notably, STE shows significant potential in generating data Ye et al. [2025] for scene text recognition (STR) Du et al. [2022], Zheng et al. [2024], Guan et al. [2023, 2025], as it can generate samples with the same style but different content from existing images.

However, it is hardly possible to get different texts with the same background style in real-world scenarios. To address this, previous methods Wu et al. [2019], Yang et al. [2020], Roy et al. [2020], Krishnan et al. [2023], Zeng et al. [2024] rely heavily on synthetic data, as shown in Fig. 1(a). On the one hand, two-stage explicit disentanglement methods Wu et al. [2019], Yang et al. [2020], Roy et al. [2020] use rendering tools to synthesize paired images with identical style but different text

*Corresponding author. E-mail: zhinchen@fudan.edu.cn

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

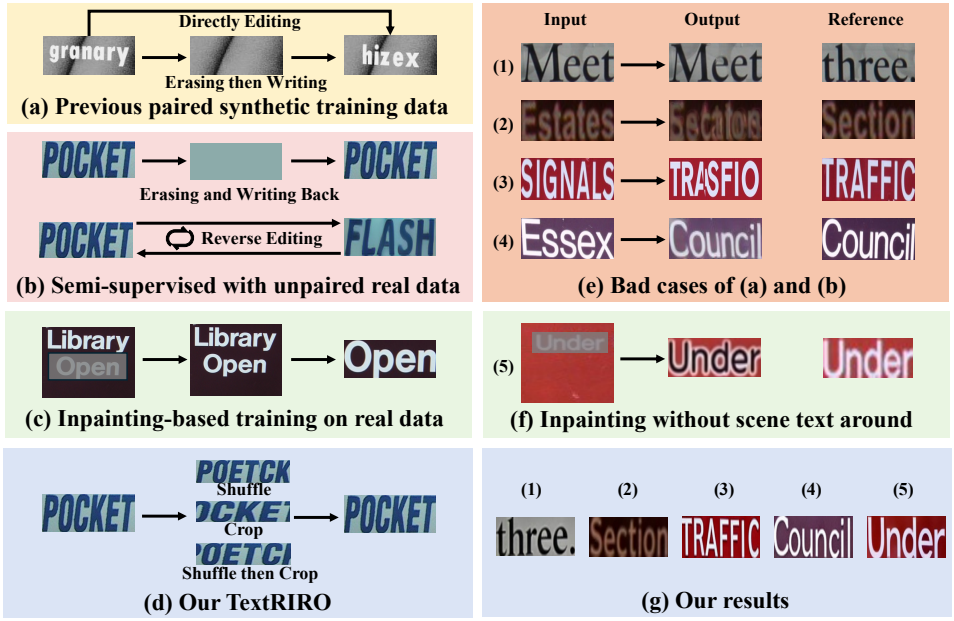


Figure 1: **A data-centric view of Scene Text Editing.** The comparison of the previous methods and TextRIRO between the data utilization paradigm (left) and the editing results (right). In (e), we present the artifacts and hallucinations of the wrong editing results in (a) and (b).

over a background image, followed by foreground-background separation and fusion. But they fail to separate text from the background, resulting in incomplete removal of the source text, as shown in cases (1) and (2) in Fig. 1(e). On the other hand, implicit disentanglement approaches Krishnan et al. [2023], Zeng et al. [2024] leverage various pre-trained models to extract features for directly generating target images, which is less controllable, resulting in hallucinations such as case (3), with misspelled or missing characters. Recent efforts Qu et al. [2023], Lee et al. [2021], Fang et al. [2025] have explored semi-supervised learning with real data by erasing itself and writing back or paired training using pseudo-labeled data, as illustrated in Fig. 1(b). Nevertheless, synthetic data still remains dominant. As evident from the editing results in Fig. 1(e), this branch of methods easily leads to case (4): even when the target text is generated, it often appears mismatched in style or contains visible artifacts. Furthermore, inpainting-based STE paradigms (Fig. 1(c)) mask the text region and attempt to restore it using surrounding context Ji et al. [2023], Chen et al. [2023a], Ye et al. [2025], Chen et al. [2026]. However, the lack of sufficient style cues makes it difficult to recover the desired appearance, as shown in Fig. 1(f).

We are motivated by an observation that prior work has under-emphasized: *the domain gap in STE is primarily a data problem, and only secondarily an architecture problem*. We therefore ask the simplest possible version of the question:

> *What if we only use real data for scene text editing, with neither synthetic triplets nor expensive paired collection?*

The core challenge is to extract stable style guidance from the source text image while suppressing the side effect from semantic text content. Our key insight is that scene text style mainly lies in low-level cues such as color, texture, and illumination, while semantic text is a high-level sequential structure. Based on this, we explore multiple self-paired real-data schemes that force the model to focus on style rather than semantics. Specifically, we experiment with (i) shuffling and recombining the text region either column-wise or block-wise (Shuffle), (ii) cropping local sub-regions (Crop), and (iii) masking parts of the original image (Mask). Empirically, we find that combining column-wise Shuffle with Crop (Shuffle&Crop) achieves the best performance.

To realize the idea, we propose **TextRIRO**, a conditional diffusion model that takes only **R**eal data as **I**nput and produces **R**ealistic **O**utput for scene **Text** editing. During training, TextRIRO applies "Shuffle&Crop" to a real text image to separate its style and semantics, and then reconstructs the

original image. By removing the reliance on synthetic data, it overcomes the bottleneck of high-quality generation during inference. For style features, we process them into multi-scale representations and then inject them into the conditional U-Net using the ControlNet-like process Zhang et al. [2023] for efficient integration. For text encoding, we use an STR pre-trained encoder Jiang et al. [2023] to extract character-level features, which improves text understanding and enables robust STE transfer.

TextRIRO removes the need for explicit or implicit disentanglement and avoids using the synthetic domain as the main training support. On real benchmarks for STE, it produces more realistic results with markedly reduced "synthetic flavor." It also achieves higher accuracy on equal-length editing and variable-length editing tasks. These results show that self-paired training on real data effectively moves STE toward realistic generation. We further ask whether STE can help STR data generation Ye et al. [2025, 2026a,c]. We find that TextRIRO already outperforms existing STE methods, but the editing accuracy still limits direct use as training data. We then use STE as data expansion by concatenating the original image with its edited counterpart. This preserves style but adds content, which yields high-quality, scene-consistent samples for long-text recognition. Starting from the labeled real dataset U14M-Filter Jiang et al. [2023], Du et al. [2025], we build a 3M-scale dataset, **TextRIRO-3M**. On Union14M-Benchmark Jiang et al. [2023], training an STR model on mixed TextRIRO-3M and real data improves overall accuracy by 1.91% over a real-only baseline, and improves the Multi-words subset by 3.28%. These gains indicate that current STR training data are not saturated and STE-based synthetic data substantially boost STR.

In summary, our main contributions are as follows:

- We propose TextRIRO, a novel STE method that trains solely on real data via self-paired supervision. It performs realistic editing without explicit or implicit disentanglement, and it alleviates artifacts and hallucinations caused by the synthetic-to-real domain gap.
- TextRIRO achieves more realistic and accurate edits on real benchmarks and attains state-of-the-art (SOTA) performance under the stronger criterion of "training with edited data."
- Built on TextRIRO, we create TextRIRO-3M, a large-scale realistic composite dataset for long-text recognition. It significantly improves STR, which validates both the realism and the utility of our edited data and demonstrates that STE effectively benefits STR.

2 Related Work

Model Perspective on STE Traditional STE methods evolve from GAN-based models Wu et al. [2019], Yang et al. [2020], Roy et al. [2020], Krishnan et al. [2023] to diffusion models Ji et al. [2023], Chen et al. [2023a], Tuo et al. [2024b], Zhao and Lian [2024], Tuo et al. [2024a], Zeng et al. [2024] and, more recently, to Unified Multimodal Models (UMMs) Zhang et al. [2024], Fang et al. [2025]. Early STE models employ GANs to directly edit foreground text, as seen in SRNet Wu et al. [2019] and its multi-stage framework for reconstructing background, rendering text, and fusing outputs. Later works expand capabilities: SwapText Yang et al. [2020] handles curved text, STEFANN Roy et al. [2020] focuses on character-level control, and TextStyleBrush Krishnan et al. [2023] leverages StyleGAN2 Karras et al. [2020] for self-supervised style-content disentanglement. Diffusion models advance STE further: DiffSTE Ji et al. [2023] uses dual encoders, DiffUTE Chen et al. [2023a] introduces inpainting and OCR-guided encoding, TextDiffuser Chen et al. [2023b] and UDiffText Zhao and Lian [2024] use segmentation masks for better generation. Frameworks like AnyText Tuo et al. [2024b] and AnyText2 Tuo et al. [2024a] unify multilingual editing via ControlNet Zhang et al. [2023]. TextCtrl Zeng et al. [2024] disentangles style conditions directly in diffusion models. Meanwhile, DARLING Zhang et al. [2024] unifies scene text editing, removal, and recognition tasks to leverage these tasks to disentangle task-specific features and RS-STE Fang et al. [2025] utilizes scene text recognition to assist scene text editing. These two approaches highlight synergy across scene text tasks.

Data Perspective on STE Model advancements in STE are closely tied to the evolution of training data. Early methods use synthetic paired data by rendering tools, e.g., SRNet-Datagen Wu et al. [2019]. To include real-world data, RewriteNet Lee et al. [2021] uses masked regions and MOSTEL Qu et al. [2023] proposes an erasing and writing-back strategy. UMM-based RS-STE Fang et al. [2025] adopts reverse self-supervision on real images. In diffusion-based STE, some models (e.g., TextCtrl Zeng et al. [2024]) disentangle features using synthetic data, while others Tuo et al. [2024b], Chen et al.

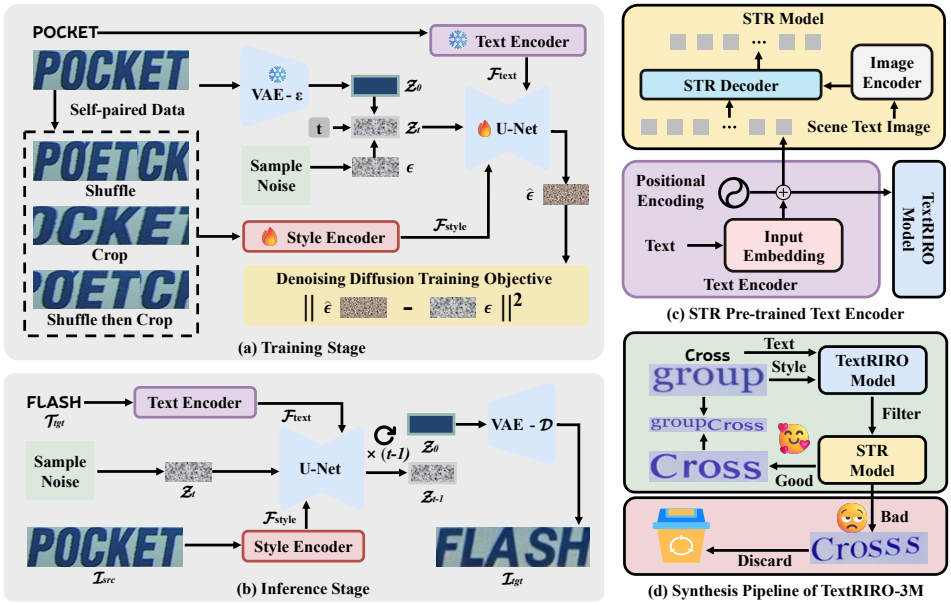


Figure 2: **TextRIRO pipeline.** (a) At training time, a single real image I_{tgt} is pushed through a content-destructive, style-preserving operator to produce a pseudo-source I_{src}. A ViT-based style encoder emits multi-scale features that are injected into the denoising U-Net through ControlNet-style zero-convolutions, while a text encoder pre-trained on *real* scene text with an autoregressive STR objective supplies the target-text conditioning. (b) At inference time, the same U-Net follows the standard SD denoising schedule. (c) Our text encoder is a lightweight character tokenizer trained jointly with an STR decoder via next-token prediction. (d) The edited image is validated by an independent STR model and, if accepted, concatenated with its original to build a style-consistent long-text sample for STR. ❄ : frozen, 🔥 : trainable.

[2023b] view STE as masked image inpainting, releasing large real datasets like Anyword-3M Tuo et al. [2024b] and MARIO-10M Chen et al. [2023b]. However, style consistency can be difficult when context lacks similar references. Benchmarks and datasets for application are also advancing: ScenePair Zeng et al. [2024] provides high-quality real-world evaluation, and TextSSR Ye et al. [2025] generates millions of edited images for STR tasks, demonstrating STE's broader utility.

3 Method

The STE process can be formally defined as follows. Denote the input source image as I_{src}, the target text as T_{tgt}, and the editing model as M. The objective is to produce the target image I_{tgt} such that:

$$I_{\text{tgt}} = M(I_{\text{src}}, T_{\text{tgt}}). \tag{1}$$

However, Eq. 1 characterizes the ideal training and inference procedure. In practice, training data rarely provide real triplets $(I_{\text{src}}, T_{\text{tgt}}, I_{\text{tgt}})$. To bridge this gap, Sec. 3.1 introduces a self-paired training strategy, Sec. 3.2 presents the unified TextRIRO architecture, and Sec. 3.3 leverages the trained model to expand STR datasets with long-text instances.

3.1 Self-Paired STE Data

Most prior data pipelines Wu et al. [2019], Qu et al. [2023] construct $(I_{\text{src}}, T_{\text{tgt}}, I_{\text{tgt}})$ triplets primarily from synthetic text images. The domain gap between synthetic and real scenes limits the ability of STE models and their applications to downstream scene text tasks. Recent methods Qu et al. [2023], Fang et al. [2025] introduce some real data for unpaired training, yet synthetic samples remain dominant and only partially bridge this gap.

When using an image as I_{src}, our goal is to remove its textual content while preserving its style. This motivates a simple strategy: scrambling the original image disrupts content while retaining stylistic

Algorithm 1 Self-Paired STE Data Generation

Require: Scene text image I_{tgt}, corresponding text label T_{tgt}, shuffle probability p_{shuffle}
Ensure: Generated self-paired source image I_{src}
Let $L = |T_{\text{tgt}}|$ denote the length of the text label.
Partition I_{tgt} along its longest axis into L equally sized segments: $\mathcal{B} = \{B_1, B_2, ..., B_L\}$.
Sample $r \sim \mathcal{U}(0, 1)$.
if $r < p_{\text{shuffle}}$ **then**
    Randomly permute the order of the segments to obtain $\mathcal{B}' = \{B'_1, ..., B'_L\}$.
else
    Set $\mathcal{B}' = \mathcal{B}$ (i.e., maintain original order).
end if
Choose $N \leq L$ and randomly select a consecutive subset of N segments from $\mathcal{B}'$.
Concatenate the selected segments to form I_{src}.
Return $(I_{\text{src}}, T_{\text{tgt}}, I_{\text{tgt}})$

features. However, since text comprises discrete characters, excessive shuffling fragments glyphs and impairs shape consistency. This presents a trade-off: we aim to maximize content removal but minimize stylistic degradation. (1) Text typically aligns along its long axis, and its content and length are known. We exploit this structural prior by dividing the image along its long axis into L (L is the number of characters) equal bins. By reordering these bins, we generate the source image I_{src} (see "Shuffle" in Fig. 2), thereby breaking semantic continuity while maintaining local appearance cues. (2) Stripe-based shuffling alone is insufficient for complex text layouts, such as curved or multi-oriented arrangements. To address this, we propose an alternative that reconstructs the image through its sub-patches. The global text arrangement is locally preserved: tilts and curvature manifest within contiguous sub-images. We form I_{src} by cropping the original image (as illustrated by "Crop" in Fig. 2), yielding self-paired data that retain the stylistic integrity of intricate layouts. (3) Combining these two strategies results in a more robust and diverse self-paired mechanism. As shown in Fig. 2 ("Shuffle then Crop"), with probability p_{shuffle}, shuffling is applied before cropping. Otherwise, cropping is applied directly to the original image. Alg. 1 presents the complete algorithm of our self-paired dataset. The subsequent ablation study presents other self-paired methods (e.g., Crop Only, Shuffle Only, Mask) and results, further demonstrating the optimality of the adopted algorithm.

3.2 Model Architecture

With the constructed self-paired triplets, TextRIRO maps the perturbed source image I_{src} and target text T_{tgt} into style and text control signals for a ControlNet-like diffusion generator Zhang et al. [2023], as shown in Fig. 2. The style encoder follows TextCtrl Zeng et al. [2024]: a ViT backbone Dosovitskiy et al. [2021] extracts image representations, and a lightweight spatial pyramid aggregates them into multi-scale features that are additively injected into the U-Net. For text, we keep character-level encoding Zhao and Lian [2024], Zeng et al. [2024] but replace synthetic glyph/CLIP pre-training with an autoregressive STR objective on real scene text Sheng et al. [2019], Jiang et al. [2023]. This improves real-domain character awareness and aligns recognition with editing.

The generator contains a VAE for latent mapping, a U-Net for denoising, and the above style/text encoders. The VAE and text encoder are frozen, while the style encoder and U-Net are jointly trained with the standard noise-prediction diffusion objective conditioned on style and text. This design is intentionally simple and backbone-agnostic: stronger diffusion backbones or conditioning modules can be substituted when more compute is available. We also evaluate a DiT-based conditional control alternative in Tab. 5, where ControlNet-style multi-scale injection remains more effective for fine-grained scene text editing. Detailed feature equations, tokenization, and the training loss are provided in Appendix Sec. A.1.

3.3 Generating Long-Text Recognition Data

After training TextRIRO, we generate a style-matched image with the same length but different content for each real input. To ensure correctness, we filter generated samples with an STR model trained on real data and keep only instances whose predicted text matches T_{tgt}. We then concatenate each filtered image with its original reference to build long-text recognition data, doubling the

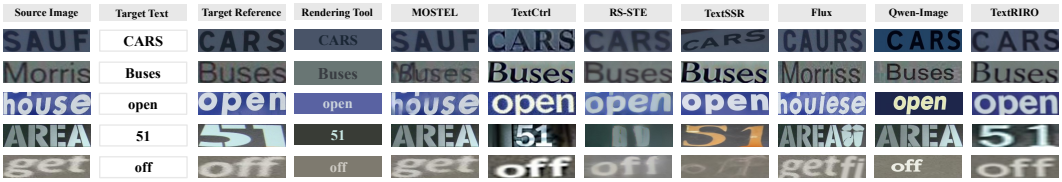


Figure 3: **Qualitative comparison on ScenePair.** "Target Reference" is the real paired ground truth. "Rendering Tool" composites the target string on the segmented foreground/background of the source and serves as an accuracy-only reference. TextRIRO is the only method that consistently reproduces the stroke thickness, perspective and illumination of the reference without leaving a synthetic residue.

sequence length. The detailed pipeline and visualizations are illustrated in Fig. 2 (d). This procedure substantially broadens coverage for rare scenarios, such as long and multi-word instances, and it establishes a practical connection between STE and STR.

4 Experiments

4.1 Editing Performance Comparison

Results on ScenePair Benchmark. We evaluate the proposed TextRIRO on the ScenePair Zeng et al. [2024] against both specialized STE methods and general editing models. The specialized baselines include the GAN-based SRNet Wu et al. [2019] and MOSTEL Qu et al. [2023], the diffusion-based inpainting methods AnyText Tuo et al. [2024b] and TextSSR Ye et al. [2025], the direct diffusion editing method TextCtrl Zeng et al. [2024], and the UMM-based RS-STE Fang et al. [2025]. The general models include Flux.1 Kontext Labs et al. [2025] and Qwen-Image-Edit Wu et al. [2025]. For these two models, we use a fixed prompt: "Edit the text to 'T_{tgt}'," where T_{tgt} is the designated target. We follow the ScenePair protocol for evaluation. For visual realism, we report FID Heusel et al. [2017], which measures differences between deep feature distributions. For text rendering correctness, we report word accuracy (ACC) and normalized edit distance (NED), computed with the official text recognizer Baek et al. [2019]. We also evaluate the recognizer on the target images.

Table 1: **ScenePair results. Boldface** indicates the best results for each metric, while underlines highlight the second best results.

Method	**ACC (%)↑**	**NED↑**	**FID↓**
SRNet 2019	17.84	0.478	49.22
MOSTEL 2023	37.69	0.557	49.19
AnyText 2024b	51.12	0.734	51.79
TextCtrl 2024	84.67	0.936	43.78
TextSSR 2025	79.69	0.924	45.37
RS-STE 2025	**91.80**	**0.981**	49.59
Flux.1 Kontext 2025	12.81	0.460	41.59
Qwen-Image-Edit 2025	53.16	0.825	49.91
TextFlux 2026	80.40	0.911	54.64
TextRIRO	85.23	0.964	**41.53**
Real (upper bound)	88.44	0.974	0.00

The quantitative results in Tab. 1 reveal that early GAN-based methods underperform both in accuracy and visual realism. When diffusion methods were introduced to the STE task, both metrics improve substantially. However, because these methods rely on synthetic data, a gap to real-data performance remains. TextRIRO leverages a self-paired real data training paradigm, achieving near-realistic accuracy and fidelity. Notably, RS-STE achieves higher text recognition accuracy, even surpassing the ground truth, but exhibits poorer FID scores, indicating that it trades realism for correctness and produces strongly synthetic-looking samples. General-purpose editing models complete part of the task, but they lag behind specialized editors, suggesting that expert STE models remain valuable.

Fig. 3 presents qualitative visual comparisons: TextRIRO clearly outperforms other methods in edit correctness, style preservation, and realism. MOSTEL, due to its two-stage editing process and training on synthetic data, does not even recognize the existence of these texts, let alone erase them. Even when it attempts to edit, it leaves residual artifacts, as in the "Buses" example. TextCtrl uses implicit disentanglement: it keeps coarse structure during feature mapping but loses fine-grained style cues, so the edited text deviates from the original style. RS-STE's samples further highlight its lack of visual authenticity and synthetic appearance, particularly for the "open" example. TextSSR, as an inpainting method, depends strongly on surrounding context: when nearby text shares a similar style (e.g., "open"), it works well, but without such references (e.g., "51"), its edits degrade. Among general-purpose editors, Flux.1 Kontext preserves the image style but rarely produces the correct target text. Qwen-Image-Edit understands the editing instruction, yet its outputs look rigid and over-synthesized, resembling results from a rendering tool.

Table 2: **RIROCommon results.** Comparison of TextRIRO with three other representative STE models. "E (Equal)" denotes text editing with equal-length substitutions, while "V (Variable)" corresponds to editing without text length limitations.

Method	**IIIT** 2012		**SVT** 2011		**IC13** 2013		**IC15** 2015		**SVTP** 2013		**CUTE** 2014		**Avg**	
	E	V	E	V	E	V	E	V	E	V	E	V	E	V
MOSTEL 2023	16.85	12.85	10.82	3.58	10.04	7.54	11.49	6.15	7.91	3.98	7.89	5.02	10.83	6.52
TextCtrl 2024	64.42	11.31	76.97	10.17	74.45	12.99	73.16	15.65	70.85	10.44	64.16	11.30	70.67	11.98
RS-STE 2025	58.80	13.21	67.08	10.73	63.71	20.11	61.29	9.10	62.17	7.59	48.39	9.62	60.24	11.73
TextRIRO (ours)	**82.74**	**50.47**	**84.39**	**46.14**	**86.23**	**53.49**	**77.91**	**45.28**	**79.69**	**42.31**	**64.87**	**48.12**	**79.31**	**47.64**

Broader Editing Evaluation. We observe that ScenePair reveals a recognition accuracy ceiling of 88.44%, largely limited by the STR model. To obtain more faithful results, we adopt a stronger and third-party text recognizer, SVTRv2 Du et al. [2025], for evaluation. TextRIRO, together with TextCtrl and RS-STE, already approaches the realism of natural images in ScenePair. Subsequent experiments therefore focus on these three methods, and we conduct an upgraded evaluation on the Common Benchmarks (hereafter RIROCommon). For each image, we provide two targets to edit: an equal-length text that matches the source length and a variable-length target text. We then report normalized accuracy like STR, where we remove non-vocabulary characters and spaces and lowercase all text before scoring.

Tab. 2 demonstrates TextRIRO's superior performance on RIROCommon: it consistently outperforms alternative models for both equal-length and variable-length editing. In particular, its variable-length editing achieves an accuracy of 47.64%, markedly higher than competitors, attributable to the inherent diversity of self-paired training examples produced via Shuffle&Crop. Conversely, RS-STE's strategy of trading off style for accuracy fails under broader and more challenging conditions. And TextRIRO sustains strong performance, achieving equal-length editing accuracy 19.07% higher than RS-STE.

Table 3: **Train-on-edits STR proxy.** "Regular" indicates the average results on the IIIT, SVT, and IC13 datasets, while "Irregular" denotes the average results on the IC15, SVTP, and CUTE datasets.

Method	IIIT 2012	SVT 2011	IC13 2013	**Regular**	IC15 2015	SVTP 2013	CUTE 2014	**Irregular**	**Avg**
MOSTEL 2023	51.87	59.51	64.99	58.79	44.62	44.34	34.72	41.22	50.01
AnyText 2024b	73.27	78.83	82.61	78.24	66.08	70.46	48.27	61.60	69.92
TextCtrl 2024	79.40	84.39	89.61	84.47	72.06	77.36	53.12	67.51	75.99
RS-STE 2025	75.43	85.32	87.98	82.91	74.43	78.45	59.72	70.87	76.89
TextRIRO (ours)	**90.53**	**89.18**	**92.65**	**90.79**	**81.00**	**84.03**	**82.64**	**82.56**	**86.67**
Real (upper bound)	99.30	97.84	98.60	98.58	90.50	96.12	98.26	94.96	96.77

STR Training Proxy Task. Although we employ a strong recognizer, it cannot reflect absolute editing accuracy. Following TextSSR Ye et al. [2025], we use the edited text to train an STR model and treat it as a proxy evaluation of the realism and accuracy of generated text. Only sufficiently realistic and accurate synthetic data will improve downstream recognition performance. This also serves as a practical measure of whether editing models can effectively generate usable training samples. We apply each STE model to U14M-Filter, editing each image with the same alternative string of equal length, and train STR models with edited images and their edited labels, then evaluate them on the STR protocol.

Tab. 3 confirms previous trends: MOSTEL's outdated architecture and data strategy result in poor performance. TextCtrl and RS-STE utilize different architectures and training data, yet achieve similar results, with only a 0.90% difference in average accuracy. Both methods follow training paradigms dominated by synthetic data, revealing the upper bound and bottleneck of such approaches, as the domain gap between synthetic and real data persists despite model optimization. In contrast, TextRIRO's self-paired training paradigm leverages real data to overcome this bottleneck, yielding an average accuracy of 86.67%, representing a 9.78% improvement over the next-best approach. This advantage is even more pronounced on irregular samples, with an 11.69% gap, highlighting TextRIRO's ability to supplement scarce training domains.



Text Enc.	Shape	VAE	Paired Op.	ACC ↑	NED ↑	FID ↓
None	[256,256]	SD 1.5	Shuffle&Crop	9.38	0.469	55.84
Glyph	[256,256]	SD 1.5	Shuffle&Crop	53.44	0.815	40.36
STR	[256,256]	SD 1.5	Shuffle&Crop	56.72	0.823	39.47
STR	[64,256]	SD 1.5	Shuffle&Crop	78.91	0.951	54.04
STR	[64,256]	SD 2.1	Shuffle&Crop	**82.42**	**0.960**	43.59
Glyph	[64,256]	SD 2.1	Shuffle&Crop	81.95	0.952	45.96
STR	[64,256]	SD 2.1	Shuffle only	45.78	0.791	39.10
STR	[64,256]	SD 2.1	Crop only	80.08	0.954	45.46
STR	[64,256]	SD 2.1	Mask	13.13	0.493	58.08
STR	[64,256]	SD 2.1	Block	79.36	0.949	50.67
STR	[64,256]	SD 2.1	None (identity)	4.37	0.176	**33.28**

Figure 4: **Ablations on ScenePair.** Left: visualizations of each self-pairing operator. Right: full factorial over text encoder pre-training, input shape, VAE variant and paired-data operator. Shuffle&Crop strictly dominates every other operator when joined with STR-pretrained text conditioning and a rectangular input. "None" collapses to an identity mapping (no perturbation operation).

4.2 Ablation Study

Contributions of each core component. Fig. 4 summarizes ablations over the TextRIRO architecture and self-paired training paradigm. Text encoding is essential: without pre-training, the model fails to learn reliable character correspondences, while STR pre-training performs better than glyph pre-training because it is learned directly from real scene text. SD 2.1 provides better VAE reconstruction than SD 1.5, and rectangular 64×256 inputs avoid the distortion caused by square resizing. For self-paired data construction, Shuffle&Crop achieves the best accuracy/realism trade-off. Crop Only lacks diversity, Shuffle Only breaks alignment, Mask destroys both content and style, Block introduces a strong distribution shift, and removing paired construction degenerates to identity copying. Fig. 5 visualizes key ablation findings. The Glyph Pre-train carries strong glyph priors and tends to lose style details, while the STR Pre-train better maintains the original style during editing. Alternative self-paired constructions also trail Shuffle&Crop in both rendering accuracy and style fidelity: Shuffle Only and Crop Only give less diversity, and Mask and Block create a large train–test gap.



Figure 5: **Qualitative ablations.** The showing setting is consistent with that in Fig. 3.

Layout-wise attribution of Shuffle and Crop. To make the complementary roles of the two perturbations explicit, we add an attribution study (in Tab. 4) on straight (IC13), curved (CUTE), and multi-oriented (SVTP) subsets, separately running Shuffle only, Crop only, and Shuffle&Crop. The reading of our design is as follows. Shuffle dominates for straight text, because it maximally breaks left-to-right semantics while preserving column-local style. Crop dominates for curved and multi-oriented text, because it preserves local tilt and curvature within contiguous sub-images.

Table 4: Layout-wise attribution of different operators. Values are equal-length editing accuracy (%).

Operator	Straight (IC13)	Curved (CUTE)	Multi-oriented (SVTP)
Shuffle only	79.00	54.83	66.67
Crop only	77.83	**60.93**	**69.76**
Shuffle&Crop	**80.16**	58.42	68.84

DiT-Based Variant. We also try to instantiate TextRIRO with DiT-based conditional control, using OminiControl Tan et al. [2025] as a representative lightweight condition-injection method. This variant can be trained under our self-paired real-data formulation, but its ScenePair performance remains behind the ControlNet-based TextRIRO. The main reason is that OminiControl-style conditioning injects image conditions in a compact token-level manner, which is parameter-efficient and general, but relatively coarse for scene text. For text editing, small local deviations can change character identity or introduce artifacts. Therefore, ControlNet's multi-scale feature injection is better suited because it preserves and transfers local style cues at several spatial resolutions. In addition, the OminiControl paper reports that, despite its minimal and universal design, it is not uniformly superior to ControlNet across all controlled generation settings. We therefore use ControlNet as the default backbone and leave more specialized DiT-based fine-grained control as future work.

Table 5: Backbone choice comparison on ScenePair Zeng et al. [2024].

Backbone	ACC (%)↑	NED↑	FID↓
OminiControl-based 2025	52.89	0.803	**39.65**
ControlNet-based 2023	**85.23**	**0.964**	41.53

Cross-recognizer evaluation. ScenePair ACC is measured with TRBA Baek et al. [2019] as Official. To test whether the observed accuracy–realism trade-off is recognizer-specific, we additionally evaluate with SVTRv2 Du et al. [2025] and PPOCR-v6-rec Zhang et al. [2026]. Tab. 6 shows the same trend for all three recognizers: TextRIRO remains close to the real-image upper bound, whereas RS-STE consistently exceeds it while having a substantially worse FID (49.59 versus 41.53). This supports the interpretation that RS-STE produces recognizer-friendly but less realistic images.

Table 6: ScenePair word accuracy (%) with three independent recognizers.

Method	Official	SVTRv2	PPOCR-v6-rec
MOSTEL 2023	37.69	33.91	37.42
TextCtrl 2024	84.67	85.08	88.67
TextSSR 2025	79.69	80.23	82.34
RS-STE 2025	**91.80**	**90.39**	**91.80**
TextRIRO	85.23	85.86	86.25
Real (upper bound)	88.44	88.67	89.14

4.3 Synthetic Data Utility

Despite TextRIRO's advances, a 10.10% accuracy gap remains compared to training with the same amount of real data. This gap mainly comes from a non-trivial number of edit failures that slip into the training set and accumulate errors. TextSSR Ye et al. [2025] filters with an STR model to select usable samples, but the retained images largely duplicate styles already present in real data, which causes distribution redundancy and limits data expansion for STR. Because STE generates text that closely matches the original style, we concatenate each original image with its filtered edited result to build TextRIRO-3M, a long-text synthetic dataset.

We mix each 1M synthetic dataset with the real dataset to compare how different synthetic sources improve recognition. The candidates include MJ Jaderberg et al. [2014] and ST Gupta et al. [2016], the diffusion-based TextSSR Ye et al. [2025], and datasets produced by the same concatenation strategy with MOSTEL Qu et al. [2023], TextCtrl Zeng et al. [2024], and RS-STE Fang et al. [2025],

Table 7: **Data quality.** Performance comparison of the STR model (MAERec Jiang et al. [2023]) with different types of synthetic data mixed with real data.

Training Data	Volume	Common Benchmarks							Union14M-Benchmark							
		IIIT	SVT	IC13	IC15	SVTP	CUTE	Avg	CUR	MLO	ART	CTL	SAL	MLW	GEN	Avg
U14M-Filter 2025 (UF)	3.22M	99.30	**97.84**	98.60	90.50	96.12	98.26	96.77	89.49	93.50	77.11	83.18	86.99	86.89	85.48	86.09
MJ 2014 + UF	1M + 3.22M	99.17	97.53	**99.18**	91.44	95.81	98.26	96.89	90.35	94.01	79.11	85.62	87.3	87.26	85.84	87.07
ST 2016 + UF	1M + 3.22M	99.33	97.53	98.95	90.94	96.74	98.26	96.96	90.44	94.08	78.00	86.14	**87.81**	87.01	85.77	87.04
TextSSR 2025 + UF	1M + 3.22M	99.07	97.53	98.83	91.28	95.50	98.61	96.80	91.06	94.52	**81.22**	86.26	87.56	88.35	85.95	87.85
MOSTEL 2023 + UF	1M + 3.22M	99.17	97.53	98.37	91.00	**96.90**	96.87	96.64	90.89	93.64	79.33	83.83	87.43	87.50	85.74	86.91
TextCtrl 2024 + UF	1M + 3.22M	99.17	**97.84**	98.48	**91.61**	95.81	98.26	96.86	90.35	94.01	79.11	84.34	87.30	87.26	85.84	86.89
RS-STE 2025 + UF	1M + 3.22M	99.10	97.06	98.6	91.39	96.74	**99.65**	97.09	90.89	94.16	80.67	85.75	87.49	86.77	85.80	87.36
TextRIRO + UF	1M + 3.22M	**99.37**	**97.84**	98.72	91.33	96.59	98.96	**97.13**	**91.18**	**94.67**	81.00	**86.65**	87.49	**89.08**	85.80	**87.98**
Real-Concat + UF	1M + 3.22M	99.27	97.53	98.95	90.50	95.97	98.61	96.81	90.89	94.59	79.67	85.24	86.73	88.23	**86.01**	87.33

which we compare against TextRIRO. Tab. 7 shows that TextRIRO yields the best improvements on both the standard Common Benchmarks and the more challenging Union14M-Benchmark. On the latter, the Multi-words subset improves by 2.19%, which aligns with the benefit of longer text introduced by concatenation. Although other editing-based generators also adopt concatenation, they remain bounded by the synthesis ceiling discussed above, and their performance still falls short of TextRIRO. Furthermore, we concatenate two independent real images to create longer sequences (Real-Concat). This yields some STR improvements, confirming that sequence-length augmentation helps. However, the performance is still lower than with TextRIRO-generated data. We attribute this gap to style inconsistency between concatenated samples.

Table 8: Extension of the experiments in Tab. 7, showing the performance of TextRIRO synthetic data on various STR models and data scales.

Model	Training Data	Volume	Common Benchmarks							Union14M-Benchmark							
			IIIT	SVT	IC13	IC15	SVTP	CUTE	Avg	CUR	MLO	ART	CTL	SAL	MLW	GEN	Avg
MAERec 2023	UF	3.22M	99.30	**97.84**	98.60	90.50	96.12	98.26	96.77	89.49	93.50	77.11	83.18	86.99	86.89	85.48	86.09
	TextRIRO + UF	1M + 3.22M	**99.37**	**97.84**	**98.72**	91.33	**96.59**	**98.96**	**97.13**	91.18	**94.67**	**81.00**	**86.65**	87.49	89.08	85.80	87.98
			+0.07	+0.00	+0.12	+0.83	+0.47	+0.70	+0.36	+1.69	+1.17	+3.89	+3.47	+0.50	+2.19	+0.32	+1.89
	TextRIRO + UF	3M + 3.22M	99.20	**97.84**	**98.72**	**91.50**	95.66	**98.96**	96.98	91.92	94.38	80.89	84.85	**87.74**	**90.17**	86.07	**88.00**
			-0.10	+0.00	+0.12	+1.00	-0.46	+0.70	+0.21	+2.43	+0.88	+3.78	+1.67	+0.75	+3.28	+0.59	+1.91
SVTRv2-S 2025	UF	3.22M	99.00	97.22	98.37	90.39	93.80	98.96	96.29	86.97	90.50	77.33	85.24	82.50	84.47	**84.28**	84.47
	TextRIRO + UF	1M + 3.22M	99.00	97.84	98.72	90.34	94.11	**99.31**	96.55	88.09	91.67	77.67	85.24	**84.31**	86.29	83.72	85.28
			+0.00	+0.62	+0.35	-0.05	+0.31	+0.35	+0.26	+1.12	+1.17	+0.34	+0.00	+1.81	+1.82	-0.56	+0.81
	TextRIRO + UF	3M + 3.22M	**99.27**	**97.99**	**98.83**	**90.89**	**94.73**	98.61	**96.72**	**89.65**	**92.26**	**78.22**	**85.49**	84.09	**86.77**	84.20	**85.81**
			+0.27	+0.77	+0.46	+0.50	+0.93	-0.35	+0.43	+2.68	+1.76	+0.89	+0.25	+1.59	+2.30	-0.08	+1.34

Tab. 8 assesses generality across models and scalability with data size. With SVTRv2-S Du et al. [2025], TextRIRO still brings consistent gains, especially on Multi-Words samples. Scaling TextRIRO to 3M further improves Union14M performance for MAERec and raises Multi-Words to 90.17%, although gains on some common subsets saturate.

5 Conclusion

In this paper, we have introduced TextRIRO, a scene text editing method that, for the first time, learns editing purely from real data through a self-paired supervision scheme. By disrupting semantic content via targeted shuffle and crop operations while maintaining style, we eliminated dependence on synthetic triplets and bypassed the need for explicit or implicit disentanglement modules. Our diffusion-based model achieves near-realistic fidelity and robust content accuracy, effectively narrowing the synthetic-to-real gap that limits previous STE approaches. Comprehensive experiments on multiple benchmarks demonstrate that TextRIRO surpasses existing STE methods in both correctness and visual realism. Additionally, we create a large-scale, realistic long-text recognition dataset, TextRIRO-3M. Training STR models on it yielded significant gains, proving that high-realism STE continues to propel STR performance beyond current saturation points. Naturally, TextRIRO excels in English thanks to abundant real scene text data, but such multilingual data is scarce. Future work will focus on accurate and realistic multilingual STE in low-resource settings.

Acknowledgments and Disclosure of Funding

This work was supported by National Natural Science Foundation of China under Grant 625B2057.

References

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 *ICCV*, pages 4714–4722, 2019.

Haoxing Chen, Zhuoer Xu, Zhangxuan Gu, Jun Lan, Xing Zheng, Yaohui Li, Changhua Meng, Huijia Zhu, and Weiqiang Wang. Diffute: Universal text editing diffusion model. In *NeurIPS*, volume 36, pages 63062–63074, 2023a.

Jingye Chen, Yupan Huang, Tengchao Lv, Lei Cui, Qifeng Chen, and Furu Wei. Textdiffuser: Diffusion models as text painters. In *NeurIPS*, volume 36, pages 9353–9387, 2023b.

Zeyu Chen, Fangmin Zhao, Yan Shu, Yichao Liu, Liu Yu, and Yu Zhou. Styletextgen: Style-conditioned multilingual scene text generation. In *CVPR*, pages 7643–7653, June 2026.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

Yongkun Du, Zhineng Chen, Caiyan Jia, Xiaoting Yin, Tianlun Zheng, Chenxia Li, Yuning Du, and Yu-Gang Jiang. SVTR: Scene text recognition with a single visual model. In *IJCAI*, pages 884–890, 2022.

Yongkun Du, Zhineng Chen, Hongtao Xie, Caiyan Jia, and Yu-Gang Jiang. Svtrv2: Ctc beats encoder-decoder models in scene text recognition. In *ICCV*, pages 20147–20156, 2025.

Yongkun Du, Zhineng Chen, Yazhen Xie, Weikang Bai, Hao Feng, Wei Shi, Yuchen Su, Can Huang, and Yu-Gang Jiang. Unirec-0.1 b: Unified text and formula recognition with 0.1 b parameters. In *ECCV*, pages 36–55, 2026.

Zhengyao Fang, Pengyuan Lyu, Jingjing Wu, Chengquan Zhang, Jun Yu, Guangming Lu, and Wenjie Pei. Recognition-synergistic scene text editing. In *CVPR*, pages 13104–13113, 2025.

Tongkun Guan, Wei Shen, Xue Yang, Qi Feng, Zekun Jiang, and Xiaokang Yang. Self-supervised character-to-character distillation for text recognition. In *ICCV*, pages 19473–19484, 2023.

Tongkun Guan, Wei Shen, and Xiaokang Yang. Ccdplus: Towards accurate character to character distillation for text recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 47(5):3546–3562, 2025. doi: 10.1109/TPAMI.2025.3533737.

Ankush Gupta, Andrea Vedaldi, and Andrew Zisserman. Synthetic data for text localisation in natural images. In *CVPR*, pages 2315–2324, 2016.

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 *NeurIPS*, volume 30, 2017.

Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Synthetic data and artificial neural networks for natural scene text recognition. *CoRR*, abs/1406.2227, 2014.

Jiabao Ji, Guanhua Zhang, Zhaowen Wang, Bairu Hou, Zhifei Zhang, Brian Price, and Shiyu Chang. Improving diffusion models for scene text editing with dual encoders. *CoRR*, abs/2304.05568, 2023.

Qing Jiang, Jiapeng Wang, Dezhi Peng, Chongyu Liu, and Lianwen Jin. Revisiting scene text recognition: A data perspective. In *ICCV*, pages 20486–20497, 2023.

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.

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.

Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *CVPR*, pages 8107–8116, 2020.

Praveen Krishnan, Rama Kovvuri, Guan Pang, Boris Vassilev, and Tal Hassner. Textstylebrush: transfer of text aesthetics from a single example. *TPAMI*, 45(7):9122–9134, 2023.

Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, Sumith Kulal, Kyle Lacey, Yam Levi, Cheng Li, Dominik Lorenz, Jonas Müller, Dustin Podell, Robin Rombach, Harry Saini, Axel Sauer, and Luke Smith. Flux.1 kontext: Flow matching for in-context image generation and editing in latent space. *CoRR*, abs/2506.15742, 2025.

Junyeop Lee, Yoonsik Kim, Seonghyeon Kim, Moonbin Yim, Seung Shin, Gayoung Lee, and Sungrae Park. Rewritenet: Reliable scene text editing with implicit decomposition of text contents and styles. *CoRR*, abs/2107.11041, 2021.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *JMLR*, 9(Nov):2579–2605, 2008.

Anand Mishra, Karteek Alahari, and CV Jawahar. Scene text recognition using higher order language priors. In *BMVC*, pages 1–11, 2012.

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.

Yadong Qu, Qingfeng Tan, Hongtao Xie, Jianjun Xu, Yuxin Wang, and Yongdong Zhang. Exploring stroke-level modifications for scene text editing. In *AAAI*, volume 37, pages 2119–2127, 2023.

Anhar Risnumawan, Palaiahankote Shivakumara, Chee Seng Chan, and Chew Lim Tan. A robust arbitrary text detection system for natural scene images. *ESWA*, 41(18):8027–8048, 2014.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, pages 10684–10695, 2022.

Prasun Roy, Saumik Bhattacharya, Subhankar Ghosh, and Umapada Pal. Stefann: scene text editor using font adaptive neural network. In *CVPR*, pages 13228–13237, 2020.

Fenfen Sheng, Zhineng Chen, and Bo Xu. NRTR: A no-recurrence sequence-to-sequence model for scene text recognition. In *ICDAR*, pages 781–786, 2019.

Zhenxiong Tan, Songhua Liu, Xingyi Yang, Qiaochu Xue, and Xinchao Wang. Ominicontrol: Minimal and universal control for diffusion transformer. In *ICCV*, pages 14940–14950, 2025.

Yuxiang Tuo, Yifeng Geng, and Liefeng Bo. Anytext2: Visual text generation and editing with customizable attributes. *CoRR*, abs/2411.15245, 2024a.

Yuxiang Tuo, Wangmeng Xiang, Jun-Yan He, Yifeng Geng, and Xuansong Xie. Anytext: Multilingual visual text generation and editing. In *ICLR*, 2024b.

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

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, et al. Qwen-image technical report. *CoRR*, abs/2508.02324, 2025.

Liang Wu, Chengquan Zhang, Jiaming Liu, Junyu Han, Jingtuo Liu, Errui Ding, and Xiang Bai. Editing text in the wild. In *ACM MM*, pages 1500–1508, 2019.

Yu Xie, Jielei Zhang, Pengyu Chen, Weihang Wang, Longwen Gao, Peiyi Li, Qian Qiao, and Zhouhui Lian. Textflux: An ocr-free dit model for high-fidelity multilingual scene text synthesis. *Computer Graphics Forum*, page e70342, 2026.

Qiangpeng Yang, Jun Huang, and Wei Lin. Swaptext: Image based texts transfer in scenes. In *CVPR*, pages 14700–14709, 2020.

Xingsong Ye, Yongkun Du, Yunbo Tao, and Zhineng Chen. Textssr: Diffusion-based data synthesis for scene text recognition. In *ICCV*, pages 17464–17473, 2025.

Xingsong Ye, Yongkun Du, JiaXin Zhang, Chen Li, Jing LYU, and Zhineng Chen. What's wrong with synthetic data for scene text recognition? a strong synthetic engine with diverse simulations and self-evolution. In *CVPR*, pages 16645–16654, 2026a.

Xingsong Ye, Yongkun Du, Jiaxin Zhang, Zhixian Li, Chong Sun, Chen Li, Jing Lyu, Lianwen Jin, and Zhineng Chen. All-in-one multilingual scene text recognition with script-aware mixture-of-experts. *CoRR*, abs/2609.24058, 2026b.

Xingsong Ye, Yongkun Du, Jiaxin Zhang, Haojie Zhang, Chong Sun, Chen Li, Jing Lyu, and Zhineng Chen. Advancing wordart-oriented scene text recognition: Datasets and methods. In *ECCV*, pages 77–94, 2026c.

Weichao Zeng, Yan Shu, Zhenhang Li, Dongbao Yang, and Yu Zhou. Textctrl: Diffusion-based scene text editing with prior guidance control. In *NeurIPS*, volume 37, pages 138569–138594, 2024.

Boqiang Zhang, Hongtao Xie, Zuan Gao, and Yuxin Wang. Choose what you need: Disentangled representation learning for scene text recognition removal and editing. In *CVPR*, pages 28358–28368, 2024.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *ICCV*, pages 3836–3847, 2023.

Yubo Zhang, Xueqing Wang, Manhui Lin, Yue Zhang, Penglongyi Deng, Ting Sun, Tingquan Gao, Zelun Zhang, Jiaxuan Liu, Changda Zhou, et al. Pp-ocrv6: From 1.5 m to 34.5 m parameters, surpassing billion-scale vlms on ocr tasks. *CoRR*, abs/2606.13108, 2026.

Yiming Zhao and Zhouhui Lian. UDiffText: A unified framework for high-quality text synthesis in arbitrary images via character-aware diffusion models. In *ECCV*, pages 217–233, 2024.

Tianlun Zheng, Zhineng Chen, Shancheng Fang, Hongtao Xie, and Yu-Gang Jiang. Cdistnet: Perceiving multi-domain character distance for robust text recognition. *Int. J. Comput. Vis.*, 132 (2):300–318, 2024.

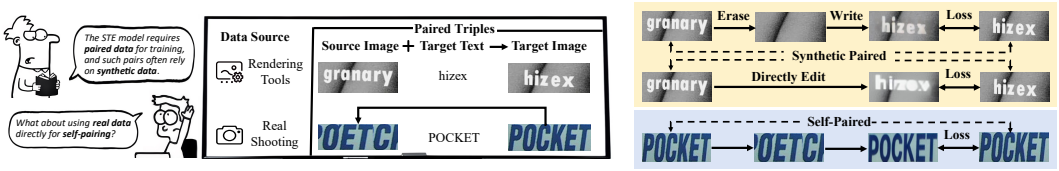


Figure 6: An interesting comic for introducing our idea. **Top:** Two previous STE training paradigms, both requiring paired synthetic data. **Bottom:** Our method leverages self-paired real data without any synthetic pairs.

A More Implementation Details

A.1 Model Formulation Details

The details of Sec. 3.2 are as follows. Given I_{src}, the style encoder first extracts ViT features

$$F_{\text{vit}} = \text{ViT}(I_{\text{src}}) \in \mathbb{R}^{N \times D}, \tag{2}$$

where N is the number of patches and D is the feature dimension. For each pyramid level k, adaptive average pooling and a zero convolution Zhang et al. [2023] produce

$$F^{(k)}_{\text{pyramid}} = \text{Conv}_{1\times1}(\text{AvgPool}_{(h_k,w_k)}(F_{\text{vit}})), \tag{3}$$

and all levels are aggregated as

$$F_{\text{style}} = \{F^{(k)}_{\text{pyramid}} \mid k = 1, ..., K\}. \tag{4}$$

The ViT takes RGB images of size 64×256, uses 16×16 patches, and outputs 768-dimensional patch embeddings. It contains 12 layers with 12 attention heads. We set $K = 4$, with pyramid output sizes $[8, 32]$, $[4, 16]$, $[2, 8]$, and $[1, 4]$.

For the text encoder, each target text T_{tgt} with length $l = |T_{\text{tgt}}|$ is truncated or padded to the maximal character length L_M. Each character t_i is mapped to a vocabulary index x_i, yielding

$$X_{\text{tgt}} = \langle \text{s} \rangle + [x_1 \dots x_l] + \langle / \text{s} \rangle + \langle \text{pad} \rangle_{l+3}^{L_M+2}, \tag{5}$$

where $\langle \text{s} \rangle$, $\langle / \text{s} \rangle$, and $\langle \text{pad} \rangle$ are special tokens. The encoded feature is

$$F_{\text{text}} = \text{Embed}(X_{\text{tgt}}) + \text{PosEnc}(X_{\text{tgt}}) \in \mathbb{R}^{(L_M+2)\times D}. \tag{6}$$

We set $L_M = 25$ and use 94 printable characters plus the space character. The embedding dimension is 768, and the text feature is fed to the conditional U-Net through cross-attention. The conditional diffusion loss is

$$\mathcal{L}_{\text{TextRIRO}} = \|\epsilon - \hat{\epsilon}(z_t, t, F_{\text{style}}, F_{\text{text}})\|^2, \tag{7}$$

where z_t is the noisy latent at timestep t, ϵ is the injected noise, and $\hat{\epsilon}$ is the predicted noise conditioned on style and text features.

A.2 Model specific settings

In the setup of Sec. 3.1 and Alg. 1, we set p_{shuffle} to 0.7 by default. To investigate the impact of this parameter, we conduct a rough ablation study with values 0.5, 0.7, and 1.0. The results are summarized in Tab. 9. We observe that setting p_{shuffle} to 0.7 yields the best overall results in terms of accuracy and NED, while maintaining competitive FID. Lower values result in reduced accuracy, whereas higher values slightly degrade FID without substantial gains in other metrics. This suggests that $p_{\text{shuffle}} = 0.7$ achieves a balanced trade-off among all evaluated criteria, and we thus adopt it as our default choice for subsequent experiments.

Table 9: Ablation study of p_{shuffle} on ScenePair Zeng et al. [2024].

p_{shuffle}	ACC (%)↑	NED↑	FID↓
0.5	80.78	0.943	**42.76**
0.7	**82.42**	**0.960**	43.59
1.0	81.72	0.947	44.01

A.3 Datasets and Implementation Details

Training Datasets. Our self-paired pipeline uses the text label and, in particular, its length, but it does *not* require character segmentation. For Shuffle, we split the image width into $n_{\text{split}} \in [2, L]$ equal-width columns, where L is the label length, and randomly permute the columns. Crop takes a horizontal contiguous sub-interval. Thus, a character may be cut by a column boundary. This is intentional because the purpose is to remove readable semantics while retaining local visual style. We train our STE model and the STR baseline on Union14M Jiang et al. [2023], currently the largest available scene text dataset. Specifically, we adopt a filtered version of its labeled subset, denoted as U14M-Filter Du et al. [2025], which contains 3.22M annotated text instances. Note that training in a completely label-free manner is theoretically infeasible, because the definition of STE inherently requires specifying the edited text.

Evaluation Datasets. To evaluate scene text editing performance under real-world conditions, we first employ the ScenePair Zeng et al. [2024], which comprises 1,280 pairs of carefully collected, real-world text images. Furthermore, we construct benchmarks for equal-length and variable-length text editing using six widely-used text recognition datasets: IIIT 5K-Words (IIIT Mishra et al. [2012]), ICDAR2013 (IC13 Karatzas et al. [2013]), Street View Text (SVT Wang et al. [2011]), ICDAR2015 (IC15 Karatzas et al. [2015]), Street View Text-Perspective (SVTP Phan et al. [2013]), and CUTE80 (CUTE Risnumawan et al. [2014]), which we collectively refer to as the Common Benchmarks. Additionally, we assess data quality using Union14M-Benchmark Jiang et al. [2023], which includes seven subsets: Curve (CUR), Multi-Oriented (MLO), Artistic (ART), Contextless (CTL), Salient (SAL), Multi-Words (MLW), and General (GEN).

Implementation Details. We use MAERec Jiang et al. [2023] both for text-encoder pre-training and as our STR proxy baseline, training for 20 epochs under the settings established by the OpenOCR framework Du et al. [2025]. TextRIRO builds upon SD 2.1 for the VAE and SD 1.5 for the U-Net backbone Rombach et al. [2022]. The extensively pre-trained VAE in SD 2.1 is well-suited for scene text tasks. The U-Net requires its parameters to be unfrozen for training, so the specific version has minimal impact. Meanwhile, SD 1.5 uses a smaller dimension ($\dim = 768$) and can directly match the feature dimension of the text encoder. All images are resized to 64×256 and those with aspect ratios less than $1:2$ are rotated 90° counterclockwise to match the target resolution. We use a batch size of 256, apply an initial learning rate of 1×10^{-5}, and train for 15 epochs on 8 NVIDIA V100 GPUs. During inference, we use 20 sampling steps and a classifier-free guidance (CFG) scale of $\omega = 2.2$. For the generation of the TextRIRO-3M dataset, we use only reference images whose original text contains at least 4 characters. Consequently, the minimum label length in TextRIRO-3M is 8. For the other STE models, we use their officially reported results when available. Otherwise, we use their released official checkpoints and the default inference hyperparameters to ensure a fair comparison.

Training–inference protocol. During training, the style condition is the perturbed view of the same real image and the target text is its original label. At inference, the source is an unperturbed real image and the target is a new string. This is an unavoidable residual mismatch because coherent real source–target pairs with different text are not available at scale. Shuffle&Crop is designed to reduce the mismatch by producing a real-image-derived style condition whose semantics are destroyed but whose local appearance remains realistic. The self-paired objective also provides a content-disentanglement constraint: the source condition has disrupted text order, while the target text is supplied only through the text encoder. If semantic content leaked through the style path, the model would copy the disrupted content rather than render the correctly ordered target. Consistently, the identity (None) ablation in Fig. 4 collapses to an ACC of 4.37, showing that an unperturbed source encourages copying instead of text-conditioned editing.

Computational Overhead and Runtime Efficiency. We report the computational requirements for both the STR pre-training and U-Net fine-tuning stages, which use 36 GPU-hours and 180 GPU-hours, respectively. For inference, we evaluate our method on a single NVIDIA V100 GPU with `diffusion_steps` set to 20 and a batch size of 32. Processing 10,000 batches requires 18 hours, corresponding to an average inference time of 0.20 seconds per image.

Table 10: Performance of different VAE variants on the quality of reconstructed scene text images.

Metrics	**VAE Version**	**IIIT** 2012	**SVT** 2011	**IC13** 2013	**IC15** 2015	**SVTP** 2013	**CUTE** 2014	**Avg**
PSNR↑	SD 1.5	25.90	30.79	28.41	29.87	28.58	26.43	28.33
	SD 2.1	27.63	**33.02**	30.23	31.99	30.68	28.30	30.31
	SD 3.5	**29.26**	32.34	**30.32**	**32.18**	**31.64**	**28.87**	**30.77**
FID↓	SD 1.5	1.36	0.50	0.50	1.22	1.83	0.50	0.98
	SD 2.1	**0.75**	**0.21**	**0.24**	**0.71**	**1.03**	**0.16**	**0.52**
	SD 3.5	1.28	0.39	0.41	1.14	1.72	0.34	0.88
Acc (%)↑	SD 1.5	98.70	97.37	97.67	89.62	92.56	97.57	95.58
	SD 2.1	98.97	97.99	98.02	89.95	93.49	**98.26**	96.11
	SD 3.5	**99.03**	**98.30**	98.25	**90.56**	94.88	97.92	**96.49**
	Real	99.30	97.84	**98.60**	90.50	**96.12**	**98.26**	96.77

VAE Ablation Study. We conduct a comprehensive ablation study by evaluating different variants of the VAE on six common STR benchmarks. Specifically, we encode and decode images using each VAE model, and then measure reconstruction quality with PSNR and FID. Furthermore, we assess the impact of VAE reconstruction on downstream text recognition by reporting Acc using a fixed recognizer. For reference, we also report the accuracy using ground-truth (unprocessed) images.

Table 11: Ablation study of CFG scale ω on ScenePair Zeng et al. [2024].

ω	**ACC (%)**↑	**NED**↑	**FID**↓
1.0	59.45	0.817	60.34
1.5	78.98	0.934	**35.37**
1.8	82.27	0.955	37.54
2.0	83.83	0.961	39.69
2.2	85.23	0.964	41.53
2.5	86.41	0.968	43.82
5.0	**88.05**	0.975	50.24
7.5	87.73	**0.976**	50.72

CFG Scale Ablation Study. We conduct an ablation study to investigate the effect of varying the CFG scale ω during inference. As shown in Tab. 11, increasing ω from 1.0 to 1.5 improves ACC and NED while substantially reducing FID from 60.34 to 35.37. Beyond $\omega = 1.5$, further increases generally improve text correctness but worsen FID, revealing a trade-off between accuracy and image fidelity. ACC reaches its highest value at $\omega = 5.0$ and decreases slightly at $\omega = 7.5$, whereas NED continues to improve. At $\omega = 2.2$, the model achieves a balanced performance, with ACC of 85.23%, NED of 0.964, and FID of 41.53. We therefore select $\omega = 2.2$ for subsequent experiments.

A.4 Additional Editing Evaluation

Per-image style fidelity beyond FID. FID is established as the evaluation metric in MOSTEL Qu et al. [2023], and TextCtrl Zeng et al. [2024] continues to use the same evaluation code to report FID on ScenePair, so we keep it for comparability. For each generated image, we compare it against its corresponding ScenePair reference image and report LPIPS (perceptual similarity), SSIM (structural similarity), and PSNR (pixel-level fidelity). Lower LPIPS and higher SSIM and PSNR indicate better agreement with the paired reference. LPIPS measures per-image perceptual and structural similarity, and FID measures how close the overall image distribution is to the reference distribution. TextRIRO achieves the best results on both in Tab. 12, which indicates that it generates images closer to the real distribution. PSNR and SSIM, in contrast, are per-image pixel-level metrics that are sensitive to small spatial shifts, anti-aliasing, and valid appearance variations in rendered text. Because the

ScenePair reference images are manually collected and not strictly pixel-aligned pairs, even a more realistic generation can score lower on these metrics simply because its pixels do not exactly match the reference.

Table 12: Per-image fidelity against the ScenePair reference and distributional fidelity. Lower LPIPS/FID and higher SSIM/PSNR are better.

Method	LPIPS↓	SSIM↑	PSNR↑	FID↓
MOSTEL 2023	0.2947	0.3311	14.05	49.19
TextCtrl 2024	0.3219	0.2610	11.68	43.78
TextSSR 2025	0.2622	0.3889	14.13	45.37
RS-STE 2025	0.2458	**0.4347**	**15.66**	49.59
TextRIRO	**0.2394**	0.3944	14.44	**41.53**

We acknowledge RS-STE's superiority on these two metrics. Methodologically, RS-STE builds L_1 and LPIPS losses directly in pixel space, whereas our diffusion-based approach builds the loss in VAE latent space, which is inherently disadvantaged at pixel-level reconstruction. Even so, TextRIRO achieves the second-best results on SSIM and PSNR, demonstrating the effectiveness of our real-data training paradigm.

A.5 Additional TextRIRO Details

Length-agnostic concatenation. The even-length regularity originates from the symmetric concatenation recipe rather than an intrinsic model limitation. We conduct an additional controlled study on long-text samples (target length at least 8) to quantify this effect. We evaluate three models on the long-text set from Multi-Words (362 odd-length and 401 even-length samples, 763 in total). The three models differ only in training data. The first is trained on real STR data (Baseline). The second adds 1M symmetric data. The third adds 0.5M symmetric and 0.5M asymmetric data (the asymmetric-augmented recipe).

Table 13: Odd- vs. even-length word accuracy (%) under different data recipes.

Training data	Odd length	Even length	Overall
Real STR data only	86.19	86.78	86.50
+ 1M symmetric (even-only)	88.12	**90.27**	**89.25**
+ 1M asymmetric-augmented	**88.67**	89.03	88.86

Two observations follow from Tab. 13. (1) With purely symmetric (even-only) concatenation, the model still generalizes to odd lengths (+1.93), which confirms that the even-length regularity is not a hard model limitation. However, a residual bias remains, as even-length gains (+3.49) exceed odd-length gains (+1.93). (2) To eliminate this bias, we introduce an asymmetric concatenation recipe. We concatenate a text with a variant of length $L - 1$, which supplies odd-length long-text samples while barely affecting edit-level supervision. Mixing 0.5M asymmetric with 0.5M symmetric samples (1M total) balances the two gains (+2.48 odd vs. +2.25 even), which removes the even-length artifact while keeping overall accuracy on par with the symmetric-only model. The length regularity is a controllable property of the data recipe.

Complementarity with synthetic sources. We also test whether TextRIRO data complements, rather than replaces, existing synthetic sources. Tab. 14 shows that adding either TextSSR or ST to TextRIRO and U14M-Filter improves the Union14M-Benchmark average, and using all sources gives the best result. TextSSR is particularly complementary for contextless text, while rendering-based ST helps curved and multi-oriented cases. Thus, our real-only construction is a choice to prioritize realistic editing, not a claim that prior synthetic data is intrinsically unusable.

Adding paired synthetic data to the editor. Our current training uses only real self-paired data. This framework can equally incorporate existing paired synthetic data. Under the ablation configuration, we add the synthetic data provided by MOSTEL Qu et al. [2023] for training.

Table 14: **Complementarity with synthetic sources.** Recognition accuracy (%) of MAERec trained with different data mixtures.

Training Data	Volume	Common Benchmarks							Union14M-Benchmark							
		IIIT	SVT	IC13	IC15	SVTP	CUTE	Avg	CUR	MLO	ART	CTL	SAL	MLW	GEN	Avg
UF (real only)	3.22M	99.30	97.84	98.60	90.50	96.12	98.26	96.77	89.49	93.50	77.11	83.18	86.99	86.89	85.48	86.09
TextSSR + UF	1M + 3.22M	99.07	97.53	**98.83**	91.28	95.50	98.61	96.80	91.06	94.52	81.22	86.26	87.56	88.35	85.95	87.85
TextRIRO + UF	1M + 3.22M	99.37	97.84	98.72	91.33	96.59	98.96	97.13	91.18	94.67	81.00	86.65	87.49	89.08	85.80	87.98
TextRIRO + TextSSR + UF	1M + 1M + 3.22M	99.30	97.99	98.60	**91.72**	96.28	98.96	97.14	91.59	94.89	81.44	**87.93**	88.19	88.83	86.22	88.44
TextRIRO + ST + UF	1M + 1M + 3.22M	99.37	97.84	98.60	91.61	**97.36**	**99.65**	97.40	**92.58**	**95.62**	81.33	86.26	87.93	**89.81**	86.22	88.54
TextRIRO + TextSSR + ST + UF	1M + 1M + 1M + 3.22M	**99.40**	**98.30**	**98.83**	91.44	97.21	99.31	**97.41**	92.25	95.47	**82.00**	87.14	**88.20**	89.44	**86.32**	**88.69**

Adding synthetic data raises text ACC and NED, which shows that synthetic supplementation brings a correctness gain. However, FID increases, which confirms that synthetic data has a real distribution gap and hurts realism when added. To preserve realism (the "Real Out" requirement), our paper adopts a real-only training strategy.

Table 15: Adding paired synthetic data to the self-paired editor under the ablation configuration.

Setting	ACC (%)↑	NED↑	FID↓
Real only (self-paired)	82.42	0.960	**43.59**
+ Synthetic data	**85.08**	**0.965**	49.34

Quality Filtering Bias. To further validate that the STR model for filtering preserves high-accuracy results and does not compromise dataset quality, we randomly sample 300 instances from the outputs generated by TextRIRO and filtered by SVTRv2 Du et al. [2025]. We then recruit three assessors, each responsible for evaluating 100 instances. The average accuracy reaches 99.33% (with individual accuracies of 99%, 99%, and 100%), demonstrating that the retained samples are highly reliable. The filter can nevertheless reject some visually correct samples when the recognizer fails on an unusual style. We choose this conservative trade-off to prioritize label correctness, and report the human check to make this selection bias explicit.

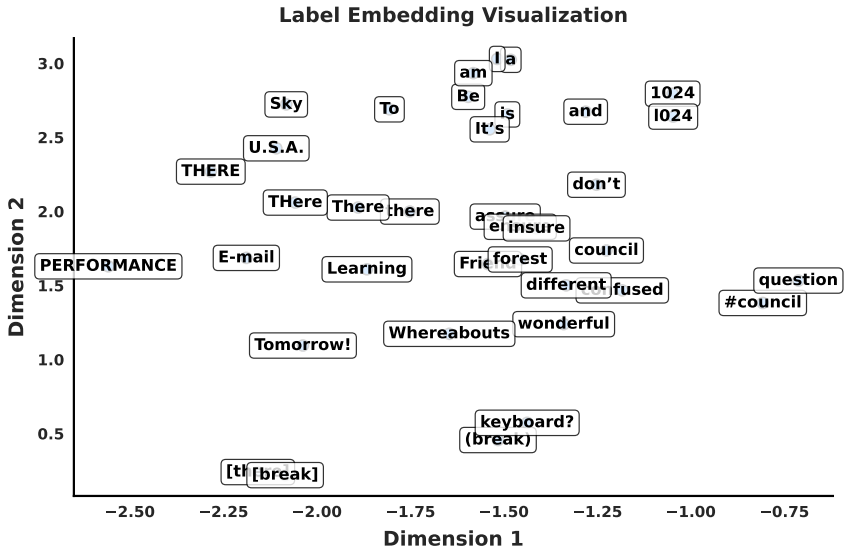


Figure 7: t-SNE Maaten and Hinton [2008] visualization of text features by STR pre-trained text encoder.

B Visualization

B.1 Character-Aware Embeddings

The STR pre-trained text encoder endows TextRIRO with strong character awareness. To further evaluate its encoding capabilities, we visualize the embeddings of several text samples using t-SNE Maaten and Hinton [2008], as illustrated in Fig. 7. Overall, words with similar character compositions or comparable text lengths are positioned closer together, while sufficient separation is maintained between distinct samples. For example, the characters '1' and 'l' possess different semantic meanings but exhibit similar glyphs, resulting in '1024' and 'l024' being embedded in close proximity. Traditionally, glyph-pre-training is required for such nuanced feature representations, yet STR pre-training achieves this directly. Additionally, the variations 'there', 'There', 'THere', and 'THERE' are embedded near each other, reflecting their shared underlying structure, yet each maintains a distinct position. This separation prevents case confusion in text generation, thereby enabling accurate character-level generation.



Figure 8: Visualization results randomly sampled from the TextRIRO-3M dataset, which doubles the text length compared to the original scene text images.

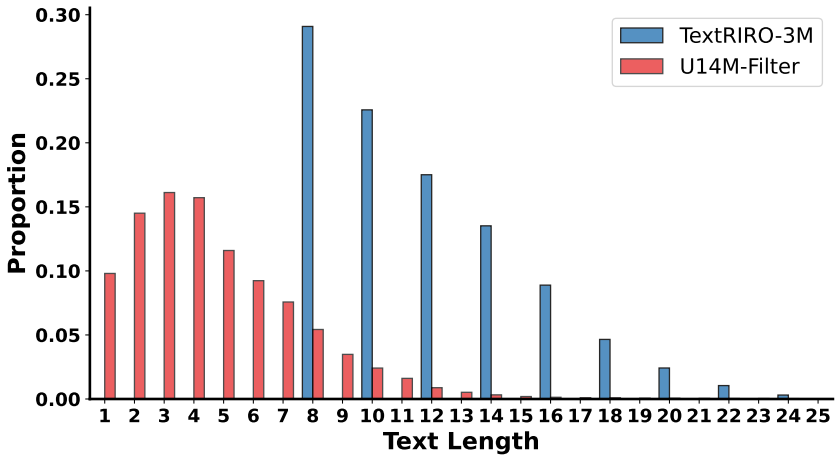


Figure 9: Length distributions of real data (U14M-Filter) and the original symmetric TextRIRO-3M construction. The symmetric recipe doubles the source length and hence produces even-length samples, and the asymmetric-augmented recipe in Tab. 13 removes this construction artifact.

B.2 TextRIRO-3M Data View

Fig. 8 presents several examples from TextRIRO-3M, demonstrating that the dataset doubles the original text length while preserving the original style. Fig. 9 further compares the label length distribution between the original symmetric TextRIRO-3M recipe and real-world data. The symmetric

construction significantly expands coverage of long text but produces even lengths only. As demonstrated in Tab. 13, asymmetric augmentation removes this artifact when broader length coverage is required.



Label:	MARISAQUEEN	semiskimmed	Tel:13877134225	Graffity-Workshop
U14M-Filter:	MANSAQUEEN	semiskimmer	Tel:1387713492 5	Groooooooooooooooo
+ *TextRIRO-1M:*	MANSAQUEEN	semiskimmed	Tel:13877134225	Graffity-Workshop
+ *TextRIRO-3M:*	MARISAQUEEN	semiskimmed	Tel:13877134225	Grocory-Workshop

Figure 10: Comparison of STR model (MAERec Jiang et al. [2023]) prediction outcomes trained on different datasets.

B.3 Enhanced STR Visualization

To clearly illustrate the effectiveness of TextRIRO-3M in improving the performance of STR models under challenging scenarios, especially for long text recognition, we visualize the prediction results of MAERec Jiang et al. [2023] trained with different datasets in Fig. 10. The experimental results show that, after incorporating TextRIRO-3M during training, the model produces more accurate and robust predictions. In particular, it demonstrates improved recognition of long and complex text sequences, effectively reducing error rates in difficult cases.

B.4 Bad Cases and Discussion

It is important to acknowledge that our STE approach is not without limitations. Fig. 11 illustrates three common failure cases, along with the corresponding causes and potential solutions:

1. **Long Text:** The text encoder used in our training pipeline is pre-trained from the STR model, which is primarily trained on datasets containing short text, resulting in limited capacity for capturing long text dependencies. Additionally, TextRIRO is trained on the same datasets, further constraining its ability to handle long text. To address this, we are inspired by the construction of the TextRIRO-3M dataset: we segment long text into shorter fragments, synthesize each fragment individually, and then concatenate these fragments to form the final long text. This approach leverages the inherent strengths of short text synthesis while enabling robust long text generation.
2. **Curved Text and Complex Geometry:** As shown in Fig. 11, TextRIRO recognizes that the input is not standard horizontally aligned text, but may interpret highly curved text as slanted. Crop preserves local tilt and curvature, and near-vertical inputs are rotated in preprocessing. Consequently, TextRIRO achieves equal-/variable-length accuracy of 64.87/48.12 on CUTE and 79.69/42.31 on SVTP, outperforming the compared baselines. Tab. 4 further isolates the operator contribution. However, extreme curvature and large geometric deformation remain difficult because regular layouts dominate the training data. Future work should incorporate finer-grained control points, similar to TPS modules in STR, to enable more precise geometric control.
3. **Special Symbols and Languages:** The synthesis of special symbols remains challenging because of both sparse training coverage and the current vocabulary/data coverage. This also restricts cross-lingual generalization: the self-pairing operation itself is language-agnostic, but sufficiently large real labeled datasets are currently much more available for Latin scripts than for low-resource languages. Collecting balanced real labeled data with broader character and language coverage is necessary for stable multilingual generation. TextRIRO will also provide more abundant synthetic data for multilingual and general OCR Ye et al. [2026b], Du et al. [2026].

Style preservation and disentanglement. Shuffle preserves stroke thickness because it permutes equal-width columns without rescaling them. Crop can change apparent stroke scale, but the two operators are complementary: pure-Shuffle instances remain in training with non-trivial probability (approximately 20% for an average label length of five) and anchor stroke-width preservation, while Crop supplies the scale variation needed for variable-length editing. The style encoder is not given an explicit transformation-type label, and it learns from the mixed perturbation distribution. Together with the disrupted semantics in the source and the separate text condition, this prevents simple

content copying, as further supported by the None ablation described above. These mechanisms do not eliminate every failure case, but explain why the combined operator is more robust than either component alone.



Figure 11: Bad Cases. We show several disappointing editing results produced by TextRIRO.

C Responsible Research Notes

Broader impacts and safeguards. TextRIRO can support benign applications such as visual content localization, accessibility, and realistic STR data generation, but scene text editing can also be misused to falsify text in images. We therefore intend to release code, data-construction scripts, and TextRIRO-3M with clear research-use and acceptable-use notices, retain STR-based quality filtering, and encourage provenance tracking or watermarking when the technology is deployed.

Existing assets and licenses. The work uses publicly available academic datasets, model components, and baselines, all cited in the main paper and bibliography. Users of released artifacts will be instructed to follow the original licenses and terms for Union14M, U14M-Filter, ScenePair, the STR benchmarks, Stable Diffusion components, and compared open-source STE models.

New assets. TextRIRO-3M is a newly constructed derived dataset. The paper documents its source data, generation pipeline, STR filtering, scale, and length distribution in Sec. 3.3, Sec. 4.2, Tab. 7, Tab. 8, and Fig. 9.

Human assessment protocol. The quality-filtering bias check in the Appendix used three assessors to judge generated text-image pairs. The instruction was: "Given a generated image and its target label, mark the sample correct only if all visible characters match the target label in order." No personal data were collected, no crowdsourcing platform was used, no external paid workers were recruited, and the task involved only visual inspection of scene-text images.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the real-data self-pairing method, TextRIRO, TextRIRO-3M, and the claimed STE and STR gains, which are supported by the method and experiments.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The appendix discusses bad cases and limitations including long text, curved text, special symbols, and low-resource multilingual settings.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper does not present theoretical theorems or formal proofs.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The method, experiments, and appendix provide the architecture, data construction, datasets, training details, and evaluation protocols needed to reproduce the main claims.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: This draft describes the implementation and states the data-release plan, but the full public data package is planned for release after acceptance.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: The appendix specifies training datasets, evaluation datasets, preprocessing, optimizer settings, epochs, batch size, image resolution, sampling steps, CFG scale, and baseline usage.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: The experiments report standard benchmark metrics but do not include error bars or statistical significance tests, mainly because the study compares large-scale deterministic benchmark evaluations.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The appendix reports GPU type, GPU-hours for text-encoder pre-training and U-Net fine-tuning, inference batch size, runtime, and per-image inference time.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The responsible research notes state that the work uses academic datasets and public baselines, identifies misuse risks, and describes intended release safeguards.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The appendix discusses positive impacts for visual content localization, accessibility, and STR data generation, as well as negative misuse risks from image text editing.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: The appendix describes acceptable-use notices, research-use release, STR-based filtering, and recommended provenance or watermarking practices for responsible use.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Existing datasets, model components, and baselines are cited, and the appendix states that released artifacts will instruct users to follow the original licenses and terms.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.

- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The new TextRIRO-3M asset is documented through its source data, generation pipeline, STR filtering, scale, and length distribution; release is planned after acceptance.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [Yes]

    Justification: The appendix provides the assessor instruction for the quality-filtering bias check and notes that no external crowdsourcing platform or personal data were used.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

Justification: The work does not conduct human-subject studies involving personal data or participant risk; the small human check is limited to visual inspection of generated scene-text images.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs are not used as an important, original, or non-standard component of the core method.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.