Signal Processing: Image Communication

journal homepage: www.elsevier.com

Unified Definition and Framework for Synthetic Text Detection and Removal

Jingru Li[a], Sheng Zhou[a], Liangcheng Li[a], Feiyu Gao[b], Jiajun Bu[a], Zhi Yu[a,**]

[a]*Zhejiang University, Hangzhou, P.R. China*
[b]*Alibaba Group, Hangzhou, P.R. China*

ABSTRACT

Practically, Synthetic Text Instance(STI) is a widely-used secondary-processed text instance in e–commercial and graphic design. STI are essential for downstream tasks like visual question answering(VQA) and document layout analysis(DLA). However, existing text detection methods retrieve all types of text instances in the image and they cannot specifically detect STI. Moreover, they can also cause misunderstanding and should be removed while keeping the consistency with the background of the image. Therefore, this paper gives a brief definition of STI and proposes a novel special network for synthetic text detection and removal, named STNet. It's composed of two architectures, i.e., an encoder-decoder structure for image structure reconstruction and synthetic text instance region prediction, and another GAN framework for finer reconstruction. We propose a series of manual benchmarks based on a well-known text detection dataset for synthetic text detection and removal tasks. Extensive experiments of both functions are conducted on the synthesized datasets, and the results demonstrate the effectiveness of our STNet.

1. Introduction

With the development of e-commercial economies, many electronic platforms like Amazon and Taobao become increasingly developing. When shopping, some advertisements contain well-designed text instances, and they're treated as *layers of image* in human inceptions or photoshop, and watermark (Hertz et al. (2019); Liu et al. (2021)) or noises in works of image denoising (Ulyanov et al. (2018); Batson and Royer (2019); Ilesanmi and Ilesanmi (2021)). They are visually inconsistent with the background of images, which can cause the misunderstanding of image. Such secondary processed text instances, or **Synthetic Text Instances(STI)**, are essential for downstream tasks like visual question answeringAntol et al. (2015) and document layout analysisBinmakhashen and Mahmoud (2019).

(a) An example with useful STI, i.e., an advertisement of a product.

(b) An example with useless STI, i.e., an image of bag with watermarks.

Fig. 1: An example of STI and real scene instances. The images are fetched by Amazon and Taobao. Green bounding boxes means ARTI and red bounding boxes means STI.

One type of STI is designed for a better representation of

**Corresponding author
e-mail: yuzhirenzhe@zju.edu.cn (Zhi Yu)

images. It is a more detailed explanation of the products in Figure 1a, for example, the title "Nonstop hydration..." is a STI. Previous text detection methods (Wang et al. (2019); Liao et al. (2020)) retrieve all possible text instances, including actual scene text instances(ASTI) like the text on the bottle in Figure 1a. ASTI can disturb the understanding of the image. Therefore, it's necessary to specifically detect STI while not detect ASTI.

Another type of STI is for anti-counterfeiting or watermark removal(Hertz et al. (2019); Liu et al. (2021)). As is shown in Figure 1b, the text attached on the bags and those floating on the image can be treated as different types of text instances because the color and texture of floating texts are not consistent with the bag in the background. Typically, such floating texts are designed for copyright protection or social communication, and they can also be treated as STI. However, such types of STI can prevent us from understanding the whole image. To repaint the area of STI while preserving consistency with original image, previous image inpainting methodsElharrouss et al. (2020) like Contextual Attention(CA)Yu et al. (2018) and RepaintLugmayr et al. (2022) are proposed. They reconstruct original image conditioned on a given mask, which lacks the ability to learn multiple multi-shape masks, which is common in STI. Therefore, it's also critical to propose a task to remove such useless STI for downstream tasks while preserving the original texts on the bag, i.e., a special method to eliminate such noises/watermarks is needed. (Adding Bounding Boxes of Fig.1)

Considering the two tasks, we propose a framework to locate, remove them and reconstruct the related background of such region, named **synthetic text detection** and **synthetic text removal**.

Recently, many text detection worksLiao et al. (2017, 2018, 2020); Wang et al. (2020, 2019) are proposed to locate the text instances in an image. Typically, they are widely used in Optical Character Recognition(OCR). Text detection intends to locate the text instance in an image. Some methods like Liu et al. (2016); Liao et al. (2017, 2018) are motivated by object detection (Ren et al. (2015); Redmon et al. (2016); Liu et al. (2016); Redmon and Farhadi (2018)). For arbitrary-shaped and curved text instances, recent methods like Baek et al. (2019); Guo et al. (2019); Liao et al. (2020) predict the segmentation of character-level instances first, and they merge the area to generate result bounding boxes. To improve the quality text detection and recognition, they use text synthesis methods to generate large-scale synthetic datasets for pretraining, like SynthTextGupta et al. (2016). Therefore, such a training strategy shows that such methods cannot classify different text instances specifically.

Considering the task of synthetic text removal, we notice that many inverse methods like watermark removal (Hertz et al. (2019); Liu et al. (2021)) and image inpainting (Yu et al. (2019, 2018); Liu et al. (2020); Lugmayr et al. (2022); Sun et al. (2020)) are proposed. Typically, given a completed image and a mask, image inpainting uses a corrupted image as the model's input in inverse problem and predicts a reconstructed image to build visually reasonable and high-quality results. It reconstructs an image's missing or occluded part to restore a more semantically continuous and visually realistic image. For synthetic text removal, STI is treated as corrupted part images as the character of synthetic text region. Besides, image inpainting methods also use a prior mask to guide the process of generationGuo et al. (2017); Sun et al. (2020). However, such methods fail to blindly reconstruct multi-region detailed texture occlusion by STI because the solution to the missing part is not unique. Therefore, it's challenging to get high-quality reconstructed results without enough guidance with mask or priors.

Considering the above challenges, we propose a **net**work for **S**ynthetic **T**ext Detection and Removal named STNet. It is a two-stage network for the proposed two tasks. The first stage focuses on synthetic text detection, and it's a UNet-like architecture to reconstruct the text regions and the structure information of reconstructed images. The second stage is motivated by contextual attention(CA) and Gated Convolution(GC) (Yu et al. (2019, 2018)), and we use a refinement network to improve the quality of the synthetic text removal.

The contributions of this paper are:

- We propose a series of manual datasets based on well-

known object detection and OCR datasets for synthetic text detection and removal tasks.

- We propose a UNet-based architecture for synthetic text detection. Furthermore, several benchmarks are applied to prove the better performance of STNet on the task than the other text detection methods.
- We finished the synthetic text removal task and compared the performance of different image inpainting modules on STNet for the task qualitatively and quantitatively.
- We explore the effect on the performance of two tasks with different synthetic ratios. The exploration proves the generalization of our method and datasets.

The paper is organized as follows. Section 2 describes the related works of text detection and image inpainting methods. Section 3 proposes a brief definition of STI and two popular tasks. Section 4 describes the details of the proposed network for the two tasks, and in Section 5 we perform different types of experiments to prove the performance of our method. Finally, in Section 6, we give a brief conclusion and future work of this paper.

2. Related Works

2.1. Text Detection Methods

Text Detection gives locations of text instances in an image. For multiple positions and large-scale image processing, many recent methods use deep learning to learn high-level representations. Deep-learning-based methods are consist of regression-based and segmentation-based methods.

The regression-based text detection methods are motivated by object detection (Ren et al. (2015); Liu et al. (2016); Redmon et al. (2016); Redmon and Farhadi (2018)). The methods intend to locate the text instances with the coordinates of bounding boxes. The methods include CTPN (Tian et al. (2016)), TextBoxes (Liao et al. (2017)), TextBoxes++ (Liao et al. (2018)), Seglink (Shi et al. (2017)), CRAFT (Baek et al. (2019)) and DRRG (Zhang et al. (2020)). They jointly optimize the regression and classification loss. They can have high efficiency, but not perform well at arbitrary-shaped text detection. Segmentation-based text detection methods are proposed. Motivated by the task of semantic segmentation (Long et al. (2015); He et al. (2017); Chen et al. (2017)), the model is optimized by the binary masks of text or non-text regions and generate the bounding boxes. Recently, methods like EAST (Zhou et al. (2017)), PSENet (Wang et al. (2019)), PixelLink (Deng et al. (2018)), CRAFT (Baek et al. (2019)), ContourNet (Wang et al. (2020)) and DBNet (Liao et al. (2020)) are proposed to deal with the task of arbitrary-shaped and curved-shaped text detection. They are widely used for large-scale scene text detection.

However, both segmentation and regression-based methods don't consider the diversity among different types of text instances. Our proposed method can only detect synthetic texts without any disturbance from real scene texts.

2.2. Image Inpainting and Object Removal

Image inpainting is a well-known image inverse problem (Ulyanov et al. (2018)). Existing image inpainting methods use different types of priors for guiding the inpainting process, like masks. According to whether use deep neural networks for guidance or not, there are two types of methods.

The traditional methods (Efros and Freeman (2001); Levin et al. (2003)) use sequential information in an image to guide the reconstruction of missing parts. Previous works like Ružić and Pižurica (2014); Isogawa et al. (2018); Guo et al. (2017), they uses patch information as the sequential signal by unfolding the image into different patches and measuring the similarity between two context image patches . Moreover, Li et al. (2017, 2016); Sridevi and Kumar (2019) are motivated by physical models, and they design diffusion models to analyze contextual information of the missing parts. However, traditional methods can only catch low-level local visual features for inference. Recently, CNN(Convolutional Neural Network)-based image inpainting methods Liu et al. (2018); Guo et al. (2019); Shin et al. (2020) build an generative architecture to reconstruct the corrupted part of the images progressively based on U-Net (Ronneberger et al. (2015)). Moreover, some methods (Zhao

et al. (2020); Liu et al. (2020); Yu et al. (2018, 2019)) uses GAN architecture to get high-quality and robust reconstruction results. Recently, Sun et al. (2020) uses edge attention maps to complete the image, and RepaintLugmayr et al. (2022) uses Diffusion Denoising Probabilistic Models(DDPMs) to progressively recover the original image.

Compared with related works, they use prior masks to guide the generation of missing parts, and such information is typically unknown in synthetic text removal task. Therefore, a method to learn the shape of masks is needed for the task.

2.3. Text Synthesis Methods

For a better training of text detection and recognition frameworks, text synthesis methods are proposed to keep the consistency between the background images and text instances. When using text synthesis methods, images with STI can be built for pretraining.

SynthText (Gupta et al. (2016)) attempts to generate STI on background images by the analysis of segmentation map and depth. Following the predicted depth map, the text instances are more properly implanted to approach the actual scenes. Recently, GAN-based methods have been proposed for more precisely 3D synthesis (Zhan et al. (2019)), but they only contain a single word in an image. VISD (Zhan et al. (2018)) uses semantic segmentation and a color scheme to keep the consistency of article styles in the background.

Recently, UnrealText (Long and Yao (2020)) proposes a 3D graphic engine to extract more information from real world. They can synthesize text instances from backgrounds with lightning, shadows, and occlusions. This paper mainly focuses on the 2D text synthesis method STNet for the feature of inconsistency and visually recognizable. We need to classify such types of text instances with real scene text instances.

3. Definition

Before processing such STI described in Section 1, we need to give some criteria.

In many published works like object removal (Shetty et al. (2018)), they customize editing the specific region in an image.

Table 1: Some explanation of symbols used in this paper.

Variables	Explanation
$\mathbf{I}$	The input image with STI.
$\mathbf{I}_{gt}$	The image without STI.
$\mathbf{I}_{co}$	The reconstructed image by TDM.
$\mathbf{I}_{fine}$	The reconstructed image by IIM.
M_{gt}	The ground truth of mask.
$\hat{\mathbf{M}}$	The predicted mask by TDM.
E_1	The TDM in STNet.
E_2	The IIM in STNet.
D	The discriminator in STNet.
$\mathcal{T}$	The region of text instances in an image.
$\mathcal{T}_s$	The region of STI in an image.

Specifically, they use two-stage methods to extract background-foreground information. Here the *foreground* object is defined as significant, evident, and single objects.

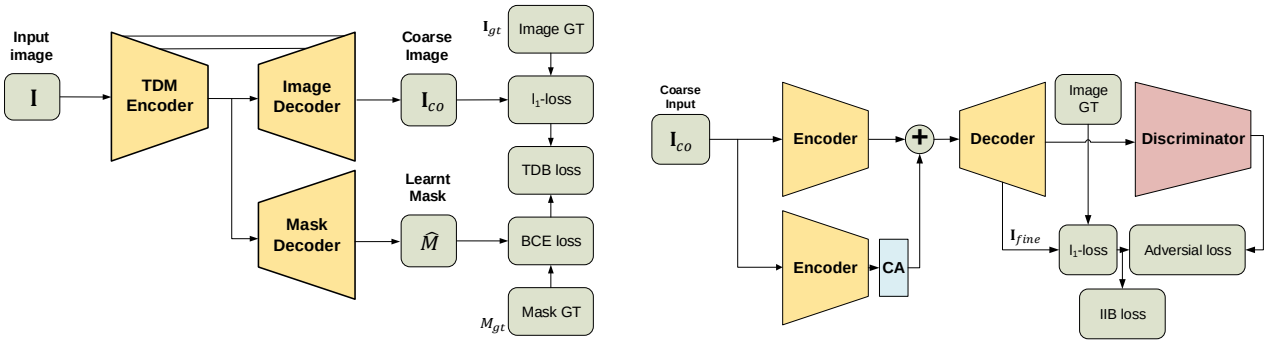
In the task of watermark removal, they treat it as a subtask of object removal, where the **foreground** object can be secondary processed motifs. Methods like Motif-Removal (Hertz et al. (2019)) use an Encoder-Decoder architecture to learn the depth and layer information of the proposed image, which can extract the layer information.

Combining the perspective of object removal and watermark removal, we can also remark the STI as an *foreground* object, i.e., text instances without any 3D distortion of image and simple geometry information. Therefore, our synthetic text detection and removal tasks are proposed to locate and remove such foreground objects.

4. Building STNet

This section focuses on synthetic text detection and removal tasks as the parallel object removal and watermark removal tasks. Unlike the object and motifs, the STI are more diverse because of their detailed texture and shapes.

From the discussion above, we can also treat synthetic text editing tasks as combining two stages, i.e., mask generation and region inpainting. Thus we can process a synthetic text instance



(a) The pipeline of TDM.

(b) The pipeline of IIM.

Fig. 2: The proposed architecture of our model. a) The whole decoder of TDM contains two branches, one for coarsely reconstructing from $\mathbf{I}$ to get $\mathbf{I}_{co}$, the other for predicting the mask $\hat{\mathbf{M}}$ of STI. The output channel of $\mathbf{I}_{co}$ is 3, and $\hat{\mathbf{M}}$ is 1 for binary prediction. b) The IIM contains a generator and a discriminator. Here the generator is a coarse-to-fine network based on the relative architecture of contextual attention Yu et al. (2018). Here we simplify the whole architecture, and we also build reconstruction loss and adversarial loss. Better viewed in color.

$t_{s,i} \in \mathcal{T}_s$ by some depth prediction methods. According to the deep image prior (Ulyanov et al. (2018)), the removal task is also an inverse problem, which can also be treated as a regularized energy minimization problem, i.e.,

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \mathbb{E}\left(f_\theta(z); x_0\right), \quad x^* = f_{\theta^*}(z) \tag{1}$$

Here $f_\theta(z)$ is typically a type of neural networks for inverse problems, like image denoising, super-resolution or image inpainting. In STNet, we formulate $E\left(f_\theta(z); x_0\right)$ as an loss function for both mask $\mathbf{M}$ reconstruction and image $\mathbf{I}$ reconstruction, where the latent variable $\mathbf{z}$ is built from original image with mask. Therefore, we design two encoder-decoder modules named **text detection module** and **image inpainting module** to estimate f_θ. Motivated by previous image inpainting and motif removal methodsYu et al. (2018); Hertz et al. (2019), we implement UNet-like architecture modules to learn the detected STI region from TDM.

Followed by the inpainting model in Yu et al. (2018, 2019), we use an UNet like architecture for TDM and SN-GAN architecture for IIM as is shown in Fig 4.

4.1. Definition

In this subsection, we intend to give a detailed symbol representation of our problem.

The variables used in this paper are defined in Table 1 For text region $\mathbf{t}$, the ground truth of bounding boxes are provided as the format $(x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4)$. For each pixel (i, j) in $\mathbf{M}$, the value is assigned as

$$M_{i,j} = \begin{cases} 1, & \text{if}(i,j) \in \mathcal{R}(\mathbf{t}) \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

Each mentioned module has its output. The detection module produces a binary mask $\hat{\mathbf{M}}$, which is the confidence to be a text of each pixel. The reconstruction branch outputs the reconstructed region of the STI $\mathbf{I}_{recon}$, and finally, the coarse or fine reconstructed image is defined as

$$\mathbf{I}_{out} = (1 - \hat{M}) \odot \mathbf{I} + \mathbf{I}_{recon} \tag{3}$$

where $\mathbf{I}_{out} \in \{\mathbf{I}_{co}, \mathbf{I}_{fine}\}$.

4.2. Text Detection Module(TDM)

The TDM E_1 pays attention to the task of synthetic text detection. The TDM focus on achieving precise location of synthetic text region $\mathcal{T}_s$.

In STNet, the TDM predict a binary mask $\hat{\mathbf{M}}$ for synthetic text region $\mathcal{T}_s$. Motivated by the architecture of Motif-Removal (Hertz et al. (2019)) and deep image prior (Ulyanov et al. (2018)), we design a U-Net like architecture, where two

parameter-shared decoders are implemented for mask and image reconstruction, presented in Figure 4. Then the learned mask $\hat{\mathbf{M}}$ predicts the confidence for the synthetic text region. After thresholding, we extract the connected component of th_s-thresholded $\hat{\mathbf{M}}$ motivated by the implementation of PSENet Wang et al. (2019) to get the set of predicted bounding boxes $\mathcal{B}$. Finally, we filter the small bounding boxes by a threshold of th_a.

Furthermore, the TDM also produces a coarse reconstructed image $\mathbf{I}_{co}$ by equation 3. Therefore, in this stage, we optimize the losses of masks

$$L_M = \frac{1}{HW}\sum_{i=1}^{H}\sum_{j=1}^{W}(M_{ij}\log\hat{M}_{ij} + (1-\hat{M}_{ij})\log(1-\hat{M}ij)) \quad (4)$$

and local reconstructed coarse images

$$L_{co} = \frac{1}{A(\mathcal{T}_s)}\|\mathbf{I}_{co} - \mathbf{I}_{gt}\|_1 \quad (5)$$

where A represents some area function. For building high-quality reconstruction results, we use the perception loss and style loss used in previous image inpainting methods (Hong et al. (2019); Yu et al. (2018)). Therefore, the final loss function for TDM L_{TDM},

$$L_{TDM} = L_M + \lambda`_{co}L_{co} + \lambda_{per}L_{per} + \lambda_{sty}L_{sty} \quad (6)$$

where L_{per} is perceptual loss and L_{sty} is style loss. In our implementation, we use VGG-16 He et al. (2016) to extract high-level feature maps of the images.

By the process of mask reconstruction, many high-level features of the image are reconstructed. As STI are secondarily processed and have different gradient signals of depth, such high-level semantic information learned by encoder-decoder architecture can be critical for distinguishing different types of text instances.

4.3. Image Inpainting Module(IIM)

To improve the quality of synthetic text removal tasks, we designed Image Inpainting Module(IIM). It's another lighter encoder-decoder architecture with contextual attention (Yu et al. (2018)) module for finer reconstruction. As is shown in many image inpainting methods (Yu et al. (2018, 2019); Zeng et al. (2020)), they use two-stage architecture to separately reconstruct the structure and texture information motivated by the idea of structure-texture decomposition (Aujol et al. (2006); Kim et al. (2018)). Specifically, they first construct the high-level structure information for coarse reconstruction and low-level texture information for finer reconstruction. Motivated by such an idea, we design the whole IIM architecture in Fig 2b.

Different from contextual attention (Yu et al. (2018)) and gated convolution (Yu et al. (2019)), the mask of the synthetic text region is **unknown** in the inference stage, since the mask $\mathbf{M}$ is needed to be learned in TDM for text detection. Luckily, a well-learned TDM can be seen as a pre-trained model for IIM to produce a $\hat{\mathbf{M}}$, and we fix the parameter in TDM and fine-tune IIM.

Due to the predicted mask $\hat{M}_{x,y} \in [0, 1]$ instead of $\{0, 1\}$, we can treat the layer as **soft** contextual attention(CA) operation. Our CA layer measures the attention score between the foreground and background patches with weighted cosine similarity for further inference as an extent of CA. The calculated score is the predicted confidence of the non-synthetic region.

The optimization of STNet is presented as Algorithm 1, which offers the main pipeline of STNet. STNet optimize TDM generator E_1, IIM generator E_2 and discriminator D separately.

4.4. Generative Inpainting model

Generative image inpainting module. To get high-quality generation results, we design a GAN-based module. We treat the whole IIMs as a generator G, and a fully convolutional network as a discriminator D to learn local information. Motivated by Yu et al. (2018, 2019), we build a discriminator with 5 downsampling convolutional layers with spectral normalization (Miyato et al. (2018)) to stabilize the training of IIM.

As a generator, we pipelined our TDM and IIM as organized by Figure 2a and Figure 2b connected by the coarse inpainting results. We put another *tanh* layer after the final convolution layer of IIM (or G) to control the output value of $\hat{\mathbf{I}}$ in the range

Fig. 3: Some images in ICDAR2015+Syn*t*x datasets. From left to right, $t = 1, 2, 3, 4$. All images contain both synthetic and real scene text instances, and they all present the diversity between different texts.

Fig. 4: The qualitative result of our TDM. Here we compare the result of 3 examples from ICDAR 2015+Syn2x. The figure includes Four comparison methods from left to right. They are PSENet (Wang et al. (2019)), textboxes++ (Liao et al. (2018)), Seglink (Shi et al. (2017)) and CRAFT (Baek et al. (2019)). The bounded results present our method for synthetic text detection. The detected bounding boxes are colored with green or red.

of $[-1, 1]$. Finally, we combine our model, and we get the loss of SN-GAN L_{GAN}.

And then we generate $\mathbf{I}_{fine}$ as the fake input, and $\mathbf{I}$ as real input. We use hinge GAN (Kavalerov et al. (2019)) to generate whole pixels progressively and generate $L_{GAN,D}$ and $L_{GAN,G}$. For L_{GAN}, we jointly combine $L_{GAN,D}$ and $L_{GAN,G}$. After training IIM with GAN, we get the finer reconstructed image $\mathbf{I}_{fine}$, and we further build the finer reconstruction loss L_{fine}, which is similar to L_{co}, here we build the loss function of IIM is

$$L_{IIM} = L_{GAN} + \lambda_{fine} L_{fine} \tag{7}$$

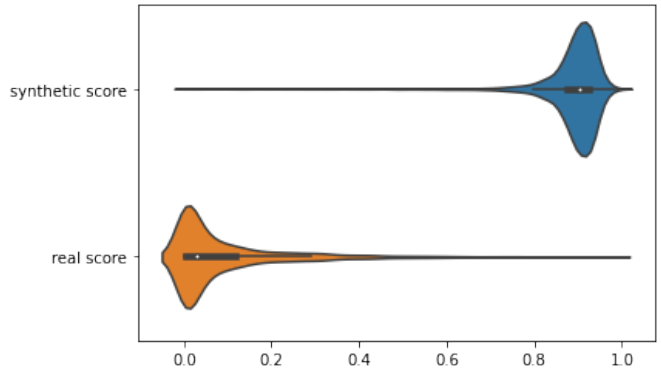
Finally, to get $\mathbf{I}_{fine}$ with higher quality, we design a iterative update algorithm to update E_1, E_2 and D separately. The detailed implementation can be referred to Algorithm 1.

5. Experiment

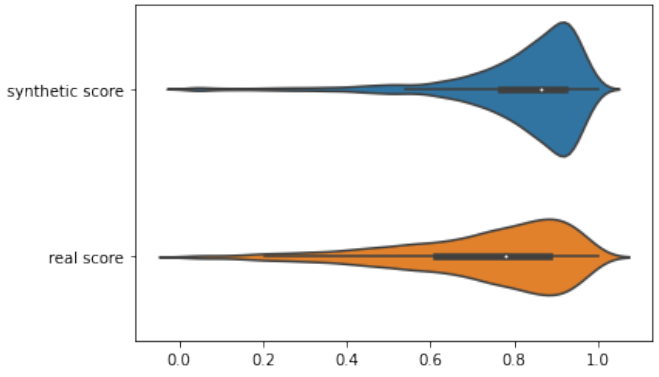
In this section, we first need to explore the performance of STNet on the task of synthetic text detection and removal. Our designed experiment attempts to answer the following questions:

Q1. How does our STNet perform on the task of synthetic text detection? Can STNet and other SOTA text detection methods clarify the distribution between different types of text instances?

Fig. 5: The qualitative performance of synthetic text detection on COCO+Syn2x. For each image, we compare the text detection results of Seglink, TextBoxes++, CRAFT, PSENet and STNet(Ours). The bounded results are predicted by our method(STNet). Both detected text instances are marked as red or green bounding boxes.



(a) Learnt text distribution by TDM of STNet. Here $s_{syn} = 0.8869 \pm 0.0431$, and $s_{real} = 0.0914 \pm 0.0990$



(b) Learnt text distribution by PSENet Wang et al. (2019). Here $s_{syn} = 0.8141 \pm 0.1137$, and $s_{real} = 0.7245 \pm 0.1637$

Fig. 6: The distribution of synthetic and actual text instances of each text region by STNet and PSENet. The upper layer represents the distribution of STI, and the bottom layer represents actual text instances. The x-axis of each figure represents the learned confidence of STI on each model.

Q2. What's the performance of synthetic text removal quantitatively and qualitatively? How do different modules of STNet on other synthetic datasets affect the performance?

Q3. What's the performance of synthetic text detection and removal among different ratios of STI on the same base dataset?

Table 2: The quantitative result for synthetic text detection compared with SOTA text detection results on ICDAR2015+Syn2x dataset. All the results in this table are achieved by ICDAR 2015 evaluation benchmark. The FPS is tested on a single Nvidia 2080 Ti GPU.

	ICDAR2015+Syn2x			ICDAR2015+Syn3x			
Metrics	Precision(↑)	Recall(↑)	H-means(↑)	Precision(↑)	Recall(↑)	H-means(↑)	FPS (↑)
Seglink	0.583	0.778	0.666	0.629	0.737	0.679	-
TextBoxes++	0.483	0.197	0.280	0.523	0.430	0.472	-
PSENet	0.635	0.757	0.691	0.686	0.756	0.720	0.35
ContourNet	0.138	0.609	0.225	0.157	0.589	0.248	-
EAST	0.653	0.589	0.619	0.671	0.654	0.663	-
PixelLink	0.606	0.670	0.636	0.131	0.122	0.127	-
CRAFT	0.636	0.788	0.704	0.683	**0.776**	0.727	**5.71**
STNet(mask)	0.823	0.797	0.810	0.803	0.714	0.756	3.23
STNet(Ours)	**0.857**	**0.808**	**0.832**	**0.838**	0.738	**0.785**	3.03

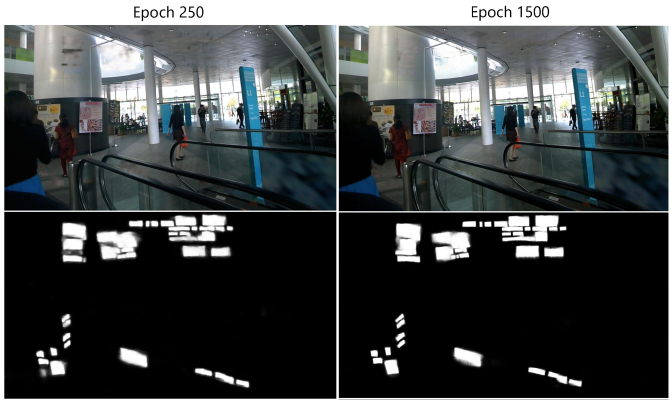


Fig. 7: The inpainting result of different epochs of IIM in different epochs of dataset ICDAR2015+Syn3x. The top line represents the reconstruction result of IIM, and the bottom line represents the predicted binary mask $\hat{\mathbf{M}}$ of synthetic text region. From left to right represents the training epoch at 250 and 1500.

5.1. Implementation Details

Our architecture is implemented by PyTorch(Paszke et al. (2019)), CuDNN 7.5, and CUDA 10.2. For the training and inference stage of TDM and IIM separately, we use one single Nvidia 2080Ti GPU. For each encoder and decoder module, we use two independent Adam optimizers with an initial learning rate of 0.0002, and we set $(0.0, 0.99)$ for the hyperparameter (α, β) of the optimizer. Here we set $\lambda_{co} = 1$, $\lambda_{mask} = 1$, $\lambda_{per} = 0.2$ and $\lambda_{sty} = 120$ for training E_1, and $\lambda_{fine} = 5$, $\lambda_D = 1$ and $\lambda_G = 0.2$ for training E_2.

We randomly crop the original image patches with resolution 192×192 for all synthetic datasets during training TDM E_1, and 128×128 for training E_2 and D. The batch size of cropped patches is 16, and we trained TDM and IIM for 50k iterations and 15k iterations. Our implementation is based on Liu et al. (2018) and Yu et al. (2019).

5.2. Datasets

We evaluate our performance of synthetic text detection and removal based on many text detection datasets, including ICDAR2015(Lucas (2005)), MSRA-TD500 (Yao et al. (2012)). For large-scale dataset, we train and evaluate our framework on a famous dataset of object detection Microsoft COCO(Lin et al. (2014)).

Building Datasets. To construct a low-bias and sufficient 2D rendering dataset, we use SynthText (Gupta et al. (2016)) to generate images **I**. SynthText predicts a dense depth map and segmentation map for each image. Moreover, they render the embedded text instance and randomly give borders and colors to the STI. Such STI are closed to Figure 1 and related to definition 1. Therefore, we use SynthText, and we adjust the ratio of repeat times for each segmented region. Here we mark the synthetic dataset as **DS+Syn*t*x**, where t is the ratio of repeat, and **DS** is the name of dataset(ICDAR2015, COCO, etc.), called **base dataset**.

Table 3: The quantitative results of synthetic text detection on COCO+Syn2x and MSRATD500+Syn2x. The compared methods are equal to those in STNet paper. We also set 0.5 for confidence and IoU threshold.

	COCO+Syn2x			MSRATD500+Syn2x		
	Precision(↑)	Recall(↑)	H-means(↑)	Precision(↑)	Recall(↑)	H-means(↑)
Seglink	0.526	0.773	0.626	0.560	0.813	0.663
TextBoxes++	0.658	0.176	0.276	0.564	0.198	0.293
EAST	0.780	0.468	0.585	0.675	0.637	0.656
PSENet	0.681	0.644	0.662	0.652	0.785	0.713
ContourNet	0.143	0.559	0.227	0.127	0.603	0.210
PixelLink	0.723	0.519	0.604	0.601	0.682	0.639
CRAFT	0.740	**0.840**	0.787	0.662	**0.877**	0.755
STNet(mask)	0.855	0.727	0.786	0.753	0.669	0.709
STNet(Ours)	**0.858**	0.733	**0.791**	**0.907**	0.808	**0.854**

This paper mainly performs on ICDAR2015 synthetic datasets since it's high-resolution, has sufficient actual text information, and is well-evaluated on different methods. Moreover, we also perform STNet on COCO for the large-scale dataset and present the generalization of STNet. The generated dataset is used to evaluate the task of synthetic text detection and removal.

To show our dataset construction procedure's performance, we provide some image instances with different synthetic repeat ratios t. The result after synthesis is presented in Fig 3, and the STI are embedded into a different region of images.

5.3. Synthetic Text Detection

Quantitative comparison. For synthetic text detection task, we provide the bounding boxes of each synthetic text instance $\mathbf{t}_i$ and further get the ground truth of region M_{gt}. For evaluation, the metric scripts provided by ICDAR 2015 and Precision/Recall/H-means are also applied for assessment. In all the evaluation results of Table 2, we use the confidence threshold of 0.9 and an area threshold of 50.

For synthetic text detection, we compare quantitative results with many other SOTA text detection methods on ICDAR2015+Syn2x and ICDAR2015+Syn3x datasets, and the results can be seen in Table 2. Both regression and segmentation based text detection methods are included. The quantitative results present our TDM can get over H-means over 0.832, with more than 10 percentage increasing of other SOTA text detection methods in H-mean value. We also compare the speed of inference on some detection models on a single Nvidia 2080Ti GPU, and we get a relatively fast speed of synthetic text detection.

From table 2, most segmentation-based text detection methods get higher Recall than Precision because these methods cannot learn the diversity between different types of text instances, and they retrieve all the possible text instances. It's reasonable because the traditional text detection methods treat the synthetic and natural scene text instances as the same type.

A more exciting discovery in Table 2 is that for most traditional methods, they can get higher Precision scores and H-means because their methods are biased to the distribution of STI. Such instances tend to be the majority of $\mathcal{T}$.

To test STNet in large-scale datasets, we also synthesize COCO+Syn2x and MSRATD500+Syn2x datasets and train them by the same settings as the above subsection mentioned. The P/R/H-means results are presented in Table 3.

Ablation study of TDM. Considering the difference between STNet and SOTA text detection methods, we mask the image reconstruction branch in TDM by only supervising the region of STI M_{gt}, and we only optimize $L_{\mathbf{M}}$ in equation 4. The result

Table 4: The ablation study of coarse synthetic text removal with different type of datasets and losses. Here "Sytle", "Perceptual" respresents the sytle loss and perceptual loss added in IIM, and "Y" represents adding the loss and "N" represents not adding.

	Losses			ICDAR2015+Syn2x		ICDAR2015+Syn3x		COCO+Syn2x	
	Style	Perceptual	Mask	PSNR(↑)	SSIM(↑)	PSNR(↑)	SSIM(↑)	PSNR(↑)	SSIM(↑)
Baseline	-	-	-	22.8009	0.9672	21.0979	0.9572	20.4695	0.8772
ResUNet(TDM)	N	N	BCE	33.4384	0.9841	33.0480	0.9820	25.9665	0.9000
ResUNet(TDM)	N	Y	BCE	34.7184	0.9858	33.3987	0.9835	25.9264	0.9006
ResUNet(TDM)	Y	Y	BCE	**34.7622**	**0.9862**	**33.7349**	**0.9840**	**26.0544**	**0.9012**

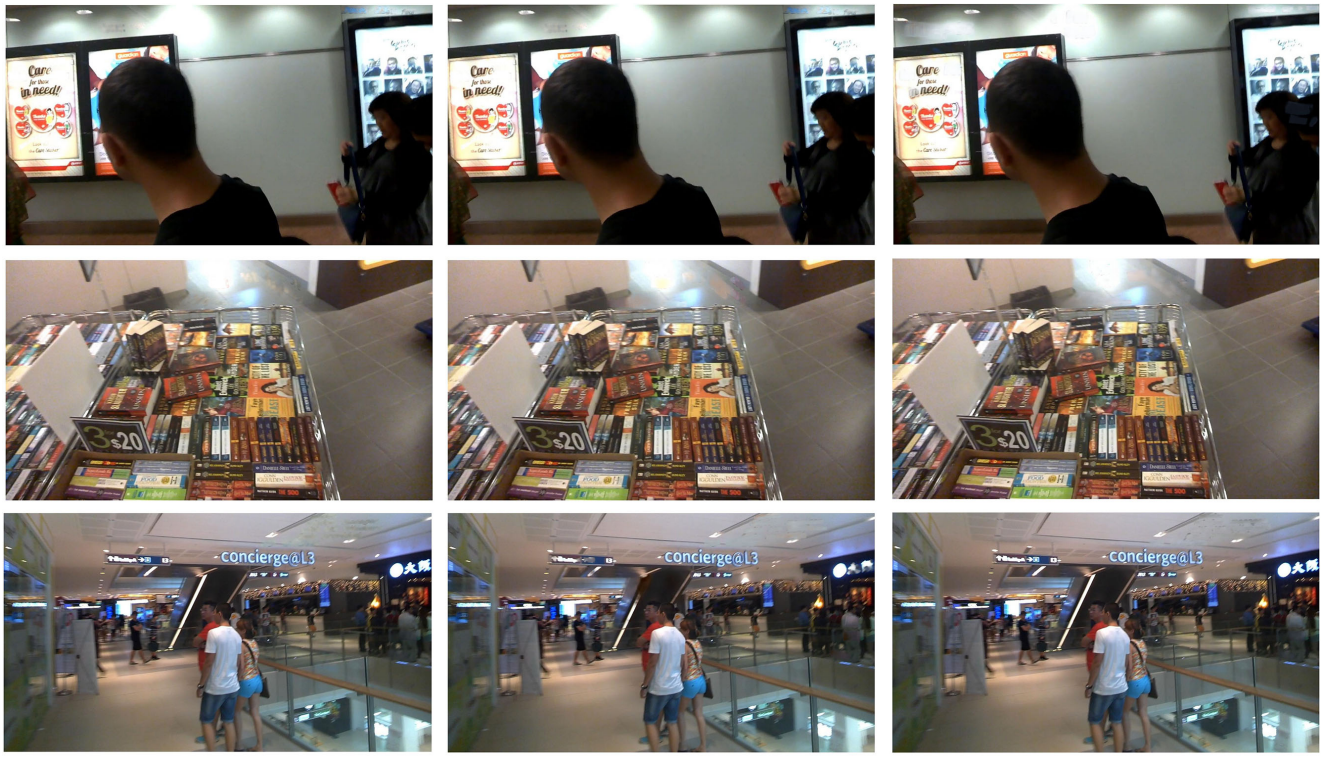


Fig. 8: The result of synthetic text removal with different IIM modules. All presented results are sampled from the validation set of ICDAR2015+Syn2x dataset. Here 'GC' represents gated convloution defined by (Yu et al. (2019)), and fine inpainting results are those produced by SN-GAN IIM modules.

is marked as *STNet(mask)* row in Table 2. With only the supervision of masks as many other text detection methods, STNet also performs well on synthetic text detection from Table 4.

Qualitative comparisons.. We visualize the same samples from ICDAR2015+Syn2x on different text detection models. Figure 4 shows that only STNet can specifically detect STI. SOTA text detection methods detect both real text instances and STI, proving that text detection methods can't learn the diversity. Our STNet can specifically detect STI, proving the ability to learn the diversity between different text instances.

Experimental analysis of TDM. To explore why our STNet can locate the STI, we need to evaluate whether TDM can learn the difference between two types of text instances. Therefore, we use the ground truth bounding boxes of real text instances($\mathcal{T}_{real}$) and STI($\mathcal{T}_s$). Then we calculate the mean confidence in each bounding box. Fig. 6 shows the visualization results of distribution between STNet and PSENet(Wang et al. (2019)). From Fig. 6a we can see the peak of each text in-

Fig. 9: The performance of synthetic text removal of images in COCO+Syn2x. From the top to the bottom, each row presents the original image $\mathbf{I}_{gt}$, the image with STI $\mathbf{I}$, the reconstructed image by TDM $\mathbf{I}_{co}$, the learnt mask $\hat{M}$ and finer reconstructed image $\mathbf{I}_{fine}$.

Table 5: The inpainting results of different types of module in IIM. "Coarse" means the coarse inpainting results of IIM, which is the reconstruction output of the first stage(using only perceptual loss). The "Coarse+GAN" line means the result after adding the refinement module.

Dataset	ICDAR2015+Syn2x		ICDAR2015+Syn3x	
Metric	PSNR(↑)	SSIM(↑)	PSNR(↑)	SSIM(↑)
Baseline	22.8009	0.9672	21.0979	0.9572
Coarse	27.1109	0.9760	26.3627	0.9721
Fine (+GAN)	**34.1791**	**0.9851**	**32.6102**	**0.9811**
Fine (+Gated)	32.6637	0.9825	32.0801	0.9807

Table 6: The quantitative result with the increasing synthetic ratio. We compare both synthetic text detection and removal results.

Ratio	1x	2x	3x	4x
PSNR(↑)	**36.9367**	34.7622	33.7369	33.5632
SSIM(↑)	**0.9913**	0.9862	0.9840	0.9840
Precision	0.774	**0.857**	0.838	0.804
Recall	0.790	**0.808**	0.738	0.701
H-means	0.782	**0.832**	0.785	0.749

stance can be clarified, while the peaks in PSENet are close enough. The distribution analysis above shows our STNet can better capture the STI.

From both qualitative and quantitative results discussed above, we have answered **Q1**.

To see the effect of TDM, we also visualize the process of the training of mask, presented in Figure 7. It shows that our TDM can fastly converge with well-learned predicted mask $\hat{\mathbf{M}}$ and $\mathbf{I}_{co}$.

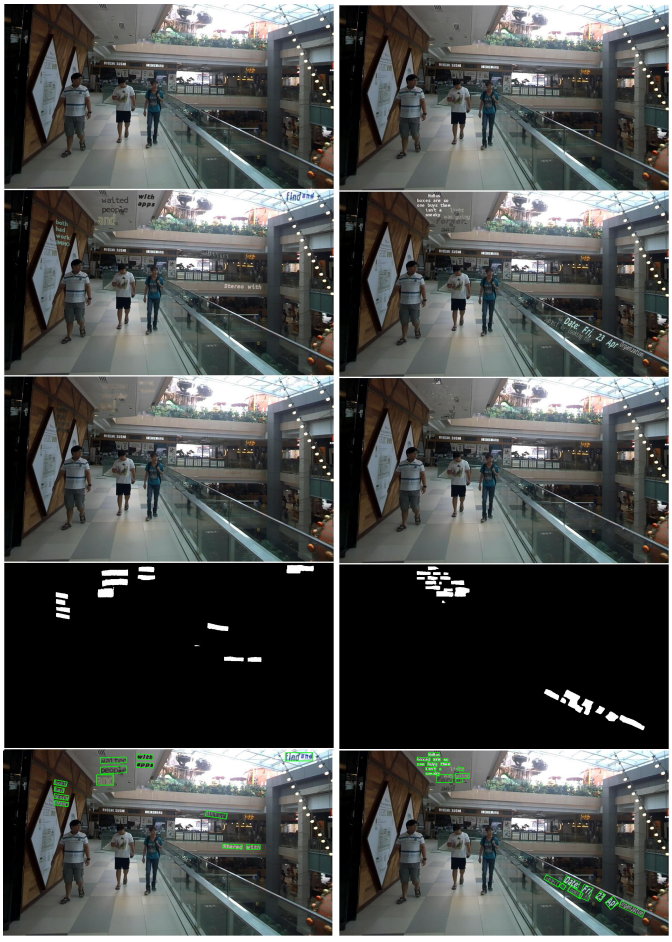
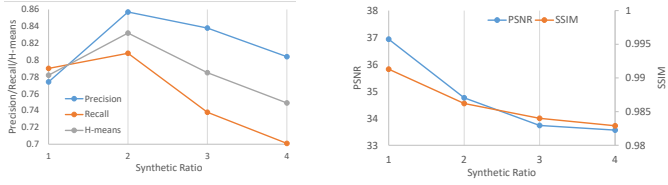


Fig. 10: Qualitative result of STNet with different synthetic ratios on dataset ICDAR2015+Syn$t$x. The learnt mask $\hat{\mathbf{M}}$ and learnt finer reconstruction result $\mathbf{I}_{fine}$ with different synthetic ratios $t = 2, 3$.

Algorithm 1 Update STNet

Require: Input masked image $\mathbf{I}_m$, ground truth of mask M_{gt}, ground truth of image $\mathbf{I}_{gt}$, critic iteration n.

1: **while** E_1, E_2, D doesn't converge **do**
2: **for** $i = 1$ to n **do**
3: $\hat{M}, \mathbf{I_{co}} \leftarrow E_1(\mathbf{I})$; // Begin update of E_1, generator for TDM.
4: $L_{mask} \leftarrow BCE(M_{gt}, \hat{\mathbf{M}})$;
5: $L_{co} \leftarrow l_1(\mathbf{I}_{co}, \mathbf{I}_{gt})$;
6: $L_{TDM} \leftarrow L_M + \lambda_{co} L_{co}$;
7: L_{TDM}.backward();
8: $\mathcal{B} \leftarrow A(ConnectedComponents(\hat{M} > th_a))$
9: $\mathbf{I}_{fine} \leftarrow \text{IIM}(\mathbf{I}_{co}, \hat{\mathbf{M}})$; // Begin update of E_2, generator of IIM.
10: $L_{fine} \leftarrow l_1(\mathbf{I}_{fine}, \mathbf{I}_{gt})$;
11: $LGAN, G \leftarrow \mathbb{E}_{\mathbf{I}_{fine} \sim p_{E_2}}(-D(\mathbf{I}_{fine}))$
12: $L_{IIM} \leftarrow L_{GAN} + \lambda L_{fine}$;
13: L_{IIM}.backward();
14: **end for**
15: Repeat line 2 and 8 to get $\mathbf{I}_{fine}$ // Update D.
16: $L_{GAN,D} \leftarrow \mathbb{E}_{\mathbf{I}_{gt} \sim p_{real}}[\max(1-D(\mathbf{I}_{gt}), 0)] + \mathbb{E}_{\mathbf{I}_{fine} \sim p_{E_2}}[\max(1-D(\mathbf{I}_{fine}), 0)]$
17: $L_{GAN,D}$.backward()
18: **end while**



(a) The synthetic text detection results with different ts.

(b) The quantative fine text removal results with different ts.

Fig. 11: The quantitative result for the task of synthetic text detection and removal on different synthetic ratio in ICDAR 2015 base dataset, where $t = 1, 2, 3, 4$ in each ICDAR2015 + Syn$t$x dataset. We mainly compares the text detection results P, R and H-means produced by TDM, and PSNR and SSIM of the inpainting results produced by IIM.

5.4. Synthetic Text Removal

Quantitative results. Similar to image inpainting, we use the same evaluation metrics as is mentioned in image inpainting

methods (Hong et al. (2019); Yu et al. (2019, 2018)), Peak-Signal-to-Noise Ratio(PSNR) and Structural Similarity Index Metric(SSIM). They are used to measure the quality and continuity of inpainting. Furthermore, the l_1 error and l_2 error between the ground truth and reconstructed image patches are widely used. In many image generation and image translation works (Karras et al. (2019, 2020)), they also use the Inception Score to measure the similarity of two datasets.

In this paper, we use PSNR and SSIM to measure the performance of synthetic text removal quantitatively for the evaluation of missing synthetic text region, for the multi-region corrupted area of original synthetic images $\mathbf{I}$.

Here table 4 compare the quantitative result of synthetic text removal task on ICDAR2015+Syn2x and ICDAR2015+Syn3x datasets. The *Baseline*s in Table 4 and Table 5 are the results by synthetic images $\mathbf{I}$. For building a better comparison benchmark, we compare coarse reconstructed image $\mathbf{I}_{co}$ trained by the method with both perceptual loss and style loss. Table 4 shows the learnt result of $\mathbf{I}_{co}$ with only E_1 trained, while 5 shows the result of end-to-end training by Algorithm 1.

Qualitative results. The qualitative results of IIM are presented in Figure 8. To make the comparison more clear, we compare different removal results with different IIM in Figure 8. Naked human eyes can hardly clarify the results.

Ablation Study. We also compare the performance of synthetic text removal by different refinement modules in IIM, including the GAN module and Gated Convolution(Yu et al. (2019)). The qualitative results are presented in Figure 8 The quantitative results are presented in Table 5, where the improvement of PSNR and SSIM values has answered **Q2**. To better compare the performance of different IIMs, we don't apply high-level losses in Table 5. Furthermore, we can get good texture similarity, and naked eyes can hardly clarify it.

5.5. Effect of different mask area of STI

For the ablation study, we explore the effect of STI on synthetic text detection and removal. Specifically, we compare different repeat ratio t in our ICDAR 2015 + Syn$t$x datasets. We visualize the generated benchmarks with different ts, where the x-axis shows different ts, and the y-axis shows the SSIM/PSNR result of reconstruction and Precision/Recall/H-means result from the detection. The results are presented in Figure 11.

We compare $t = 1, 2, 3, 4$ in ICDAR+Syn$t$x dataset. For each t, we train our model on the mixed dataset with different ts. We first optimize E_1 until convergence, and finetune E_2 and D until convergence. Finally, we evaluate on the dataset with each t.

With the increasing number of t, the performance of synthetic text removal gradually decreases monotonically. It's reasonable because the rising corrupted area will prevent the completion of the image. Moreover, when $t = 2$, we can get the best synthetic text detection performance for synthetic data. Therefore, **Q3** has been answered by the above results, and our model can handle different ratios of STI. To get a more theoretical analysis, we can treat t as the ratio of mask region in synthetic text removal.

6. Conclusion

This paper proposes the definition of widely used STI and concludes some. We also build an architecture to generate the mask of synthetic text region and the fine-grained reconstructed images. Besides, we also propose datasets with STI for the above tasks. For further discussion, except for SynthText (Gupta et al. (2016)), a synthetic method with more diversity is needed, and we also provide more results on different types of synthetic datasets.

In the future, we expect to extend our work to object deocclusion and build some theoretical analysis of diversity between STI and real scene text instances. Advertisements like Figure 1 are human-designed. They contain low semantically continuity, which is not consistent with image inpainting. A better method for such type of data is necessary for further work. For future design, it's necessary to build a more general method for different types of datasets.

7. Author Contribution

- **Jingru Li:** Conceptualization, Methodology, Validation, Formal analysis, Investigation, Writing - Original Draft,

Software, Visualization.

- **Sheng Zhou:** Validation, Formal analysis, Investigation, Resources, Writing - Review & Editing, Supervision.
- **Liangcheng Li:** Foraml analysis, Resources, Writing - Review & Editing, Supervision, Data Curation.
- **Feiyu Gao:** Methodology, Validation, Supervision, Data Curation.
- **Jiajun Bu:** Supervision, Project administration, Funding acquisition.
- **Zhi Yu:** Supervision, Project administration, Funding acquisition, Writing - Review & Editing, Software, Data Curation.

8. Acknowledgement

This work is supported by the National Key R&D Program of China (No. 2019YFF0302601) and the Alibaba-Zhejiang University Joint Institute of Frontier Technologies.

References

Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Zitnick, C.L., Parikh, D., 2015. Vqa: Visual question answering, in: Proceedings of the IEEE international conference on computer vision, pp. 2425–2433.

Aujol, J.F., Gilboa, G., Chan, T., Osher, S., 2006. Structure-texture image decomposition—modeling, algorithms, and parameter selection. International journal of computer vision 67, 111–136.

Baek, Y., Lee, B., Han, D., Yun, S., Lee, H., 2019. Character region awareness for text detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 9365–9374.

Batson, J., Royer, L., 2019. Noise2self: Blind denoising by self-supervision, in: International Conference on Machine Learning, PMLR. pp. 524–533.

Binmakhashen, G.M., Mahmoud, S.A., 2019. Document layout analysis: a comprehensive survey. ACM Computing Surveys (CSUR) 52, 1–36.

Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L., 2017. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE transactions on pattern analysis and machine intelligence 40, 834–848.

Deng, D., Liu, H., Li, X., Cai, D., 2018. Pixellink: Detecting scene text via instance segmentation, in: Thirty-second AAAI conference on artificial intelligence.

Efros, A.A., Freeman, W.T., 2001. Image quilting for texture synthesis and transfer, in: Proceedings of the 28th annual conference on Computer graphics and interactive techniques, pp. 341–346.

Elharrouss, O., Almaadeed, N., Al-Maadeed, S., Akbari, Y., 2020. Image inpainting: A review. Neural Processing Letters 51, 2007–2028.

Guo, Q., Gao, S., Zhang, X., Yin, Y., Zhang, C., 2017. Patch-based image inpainting via two-stage low rank approximation. IEEE transactions on visualization and computer graphics 24, 2023–2036.

Guo, Z., Chen, Z., Yu, T., Chen, J., Liu, S., 2019. Progressive image inpainting with full-resolution residual network, in: Proceedings of the 27th ACM International Conference on Multimedia, pp. 2496–2504.

Gupta, A., Vedaldi, A., Zisserman, A., 2016. Synthetic data for text localisation in natural images, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2315–2324.

He, K., Gkioxari, G., Dollár, P., Girshick, R., 2017. Mask r-cnn, in: Proceedings of the IEEE international conference on computer vision, pp. 2961–2969.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778.

Hertz, A., Fogel, S., Hanocka, R., Giryes, R., Cohen-Or, D., 2019. Blind visual motif removal from a single image, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6858–6867.

Hong, X., Xiong, P., Ji, R., Fan, H., 2019. Deep fusion network for image completion, in: Proceedings of the 27th ACM International Conference on Multimedia, pp. 2033–2042.

Ilesanmi, A.E., Ilesanmi, T.O., 2021. Methods for image denoising using convolutional neural network: a review. Complex & Intelligent Systems 7, 2179–2198.

Isogawa, M., Mikami, D., Iwai, D., Kimata, H., Sato, K., 2018. Mask optimization for image inpainting. IEEE Access 6, 69728–69741.

Karras, T., Laine, S., Aila, T., 2019. A style-based generator architecture for generative adversarial networks, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4401–4410.

Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T., 2020. Analyzing and improving the image quality of stylegan, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8110–8119.

Kavalerov, I., Czaja, W., Chellappa, R., 2019. Cgans with multi-hinge loss. arXiv preprint arXiv:1912.04216 .

Kim, Y., Ham, B., Do, M.N., Sohn, K., 2018. Structure-texture image decomposition using deep variational priors. IEEE Transactions on Image Processing 28, 2692–2704.

Levin, A., Zomet, A., Weiss, Y., 2003. Learning how to inpaint from global image statistics, in: null, IEEE. p. 305.

Li, H., Luo, W., Huang, J., 2017. Localization of diffusion-based inpainting in digital images. IEEE Transactions on Information Forensics and Security 12, 3050–3064.

Table 7: The statistical results of different synthetic datasets used in this paper. Here "Num Boxes" represents the number of bounding boxes in this dataset.

	Train		Test	
	Num Images	Num Boxes	Num Images	Num Boxes
ICDAR2015+Syn1x	1000	9833	500	4668
ICDAR2015+Syn2x	1000	16846	500	8597
ICDAR2015+Syn3x	1000	22588	500	11649
ICDAR2015+Syn4x	1000	27145	500	13424
MSRATD500+Syn2x	300	4897	200	3222
COCO+Syn2x	111978	879029	4796	40708

Li, K., Wei, Y., Yang, Z., Wei, W., 2016. Image inpainting algorithm based on tv model and evolutionary algorithm. Soft Computing 20, 885–893.

Liao, M., Shi, B., Bai, X., 2018. Textboxes++: A single-shot oriented scene text detector. IEEE transactions on image processing 27, 3676–3690.

Liao, M., Shi, B., Bai, X., Wang, X., Liu, W., 2017. Textboxes: A fast text detector with a single deep neural network, in: Thirty-First AAAI Conference on Artificial Intelligence.

Liao, M., Wan, Z., Yao, C., Chen, K., Bai, X., 2020. Real-time scene text detection with differentiable binarization., in: AAAI, pp. 11474–11481.

Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L., 2014. Microsoft coco: Common objects in context, in: European conference on computer vision, Springer. pp. 740–755.

Liu, G., Reda, F.A., Shih, K.J., Wang, T.C., Tao, A., Catanzaro, B., 2018. Image inpainting for irregular holes using partial convolutions, in: Proceedings of the European Conference on Computer Vision (ECCV), pp. 85–100.

Liu, H., Jiang, B., Song, Y., Huang, W., Yang, C., 2020. Rethinking image inpainting via a mutual encoder-decoder with feature equalizations. arXiv preprint arXiv:2007.06929 .

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C., 2016. Ssd: Single shot multibox detector, in: European conference on computer vision, Springer. pp. 21–37.

Liu, Y., Zhu, Z., Bai, X., 2021. Wdnet: Watermark-decomposition network for visible watermark removal, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 3685–3693.

Long, J., Shelhamer, E., Darrell, T., 2015. Fully convolutional networks for semantic segmentation, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 3431–3440.

Long, S., Yao, C., 2020. Unrealtext: Synthesizing realistic scene text images from the unreal world. arXiv preprint arXiv:2003.10608 .

Lucas, S.M., 2005. Icdar 2005 text locating competition results, in: Eighth International Conference on Document Analysis and Recognition (ICDAR'05), IEEE. pp. 80–84.

Lugmayr, A., Danelljan, M., Romero, A., Yu, F., Timofte, R., Van Gool, L., 2022. Repaint: Inpainting using denoising diffusion probabilistic models. arXiv preprint arXiv:2201.09865 .

Miyato, T., Kataoka, T., Koyama, M., Yoshida, Y., 2018. Spectral normalization for generative adversarial networks. arXiv preprint arXiv:1802.05957 .

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S., 2019. Pytorch: An imperative style, high-performance deep learning library, in: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (Eds.), Advances in Neural Information Processing Systems 32. Curran Associates, Inc., pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-pdf.

Redmon, J., Divvala, S., Girshick, R., Farhadi, A., 2016. You only look once: Unified, real-time object detection, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 779–788.

Redmon, J., Farhadi, A., 2018. Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767 .

Ren, S., He, K., Girshick, R., Sun, J., 2015. Faster r-cnn: Towards real-time object detection with region proposal networks, in: Advances in neural information processing systems, pp. 91–99.

Ronneberger, O., Fischer, P., Brox, T., 2015. U-net: Convolutional networks for biomedical image segmentation, in: International Conference on Medical image computing and computer-assisted intervention, Springer. pp. 234–241.

Ružić, T., Pižurica, A., 2014. Context-aware patch-based image inpainting using markov random field modeling. IEEE transactions on image processing 24, 444–456.

Shetty, R., Fritz, M., Schiele, B., 2018. Adversarial scene editing: Automatic object removal from weak supervision. arXiv preprint arXiv:1806.01911 .

Shi, B., Bai, X., Belongie, S., 2017. Detecting oriented text in natural images by linking segments, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2550–2558.

Shin, Y.G., Sagong, M.C., Yeo, Y.J., Kim, S.W., Ko, S.J., 2020. Pepsi++: Fast and lightweight network for image inpainting. IEEE Transactions on Neural Networks and Learning Systems .

Sridevi, G., Kumar, S.S., 2019. Image inpainting based on fractional-order nonlinear diffusion for image reconstruction. Circuits, Systems, and Signal Processing 38, 3802–3817.

Sun, L., Zhang, Q., Wang, W., Zhang, M., 2020. Image inpainting with learnable edge-attention maps. IEEE Access 9, 3816–3827.

Tian, Z., Huang, W., He, T., He, P., Qiao, Y., 2016. Detecting text in natural image with connectionist text proposal network, in: European conference on computer vision, Springer. pp. 56–72.

Ulyanov, D., Vedaldi, A., Lempitsky, V., 2018. Deep image prior, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 9446–9454.

Wang, W., Xie, E., Li, X., Hou, W., Lu, T., Yu, G., Shao, S., 2019. Shape robust text detection with progressive scale expansion network, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 9336–9345.

Wang, Y., Xie, H., Zha, Z.J., Xing, M., Fu, Z., Zhang, Y., 2020. Contournet: Taking a further step toward accurate arbitrary-shaped scene text detection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 11753–11762.

Yao, C., Bai, X., Liu, W., Ma, Y., Tu, Z., 2012. Detecting texts of arbitrary orientations in natural images, in: 2012 IEEE conference on computer vision and pattern recognition, IEEE. pp. 1083–1090.

Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., Huang, T.S., 2018. Generative image inpainting with contextual attention, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 5505–5514.

Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., Huang, T.S., 2019. Free-form image inpainting with gated convolution, in: Proceedings of the IEEE International Conference on Computer Vision, pp. 4471–4480.

Zeng, Y., Lin, Z., Yang, J., Zhang, J., Shechtman, E., Lu, H., 2020. High-resolution image inpainting with iterative confidence feedback and guided upsampling, in: European Conference on Computer Vision, Springer. pp. 1–17.

Zhan, F., Lu, S., Xue, C., 2018. Verisimilar image synthesis for accurate detection and recognition of texts in scenes, in: Proceedings of the European Conference on Computer Vision (ECCV), pp. 249–266.

Zhan, F., Zhu, H., Lu, S., 2019. Spatial fusion gan for image synthesis, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3653–3662.

Zhang, S.X., Zhu, X., Hou, J.B., Liu, C., Yang, C., Wang, H., Yin, X.C., 2020. Deep relational reasoning graph network for arbitrary shape text detection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9699–9708.

Zhao, L., Mo, Q., Lin, S., Wang, Z., Zuo, Z., Chen, H., Xing, W., Lu, D., 2020. Uctgan: Diverse image inpainting based on unsupervised cross-space translation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5741–5750.

Zhou, X., Yao, C., Wen, H., Wang, Y., Zhou, S., He, W., Liang, J., 2017. East: an efficient and accurate scene text detector, in: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pp. 5551–5560.

Supplementary Material

8.1. Detail of Synthetic Datasets

In this section, we propose some synthetic datasets based on SynthText. Here we calculate the number of images and STI in each dataset. We have built 6 datasets, with different synthetic ratios $t = 1, 2, 3, 4$ for ICDAR 2015, and set $t = 2$ for dataset MSRA-TD500 and COCO. The statistic results are shown in Table 7

8.2. More Implementation Details

Different designs of IIB. Our implementation is based on Algorithm 1. Here we propose three candidate IIBs, and the detailed layers for each IIB are presented in Table hahaha. For *Coarse2Fine* model, we use more blocks to build deeper architecture. *CA* represents the contextual attention layer, which treats the input image $\mathbf{I}$ as background and masked input coarse image $\mathbf{I}_{co}$ as foreground.

The *GatedTinyCoarse2Fine* replaces all convolutional layers with gated convolutional layers(GC). Assume the l-th layer of feature map is $F_l \in \mathbb{R}^{H\times W\times C}$, then the gated convolution layer represents:

$$\begin{aligned} g_{l+1} &= W_g * F_l[:, :, : C//2] \\ f_{l+1} &= W_f * F_l[:, :, C//2 :] \\ F_{l+1} &= \phi(f_{l+1}) \odot \sigma(g_{l+1}) \end{aligned} \tag{8}$$

where ϕ represents an activation function (such as ReLU, ELU, LReLU, etc.), and σ represents *sigmoid* function. The gated convolutional layer learns the importance of selection for each channel and spatial information.

Different choices of GAN. In our implementation, we choose SN-GAN, where spectral normalization is used in our discriminator. We also implement LSGAN and SN-PatchGAN, but we choose SN-GAN as it reaches the highest performance.

$$
\begin{aligned}
\mathcal{L}_G &= -\mathbb{E}_{z\sim\mathbb{P}_z(z)}[D^{sn}(G(z))] \\
\mathcal{L}_{D^{sn},real} &= \mathbb{E}_{x\sim\mathbb{P}_{\text{data}}(x)}[\text{ReLU}(1 - D^{sn}(x))] \\
\mathcal{L}_{D^{sn},fake} &= \mathbb{E}_{z\sim\mathbb{P}_z(z)}[\text{ReLU}(1 + D^{sn}(G(z)))] \\
\mathcal{L}_{D^{sn}} &= \mathcal{L}_{D^{sn},real} + \mathcal{L}_{D^{sn},fake}
\end{aligned}
\tag{9}
$$