Typography with Decor: Intelligent Text Style Transfer

Wenjing Wang, Jiaying Liu*, Shuai Yang, and Zongming Guo
Institute of Computer Science and Technology, Peking University, Beijing, China

Abstract

Text effects transfer can dramatically make the text visually pleasing. In this paper, we present a novel framework to stylize the text with exquisite decor, which are ignored by the previous text stylization methods. Decorative elements pose a challenge to spontaneously handle basal text effects and decor, which are two different styles. To address this issue, our key idea is to learn to separate, transfer and recombine the decors and the basal text effect. A novel text effect transfer network is proposed to infer the styled version of the target text. The stylized text is finally embellished with decor where the placement of the decor is carefully determined by a novel structure-aware strategy. Furthermore, we propose a domain adaptation strategy for decor detection and a one-shot training strategy for text effects transfer, which greatly enhance the robustness of our network to new styles. We base our experiments on our collected topography dataset including 59,000 professionally styled text and demonstrate the superiority of our method over other state-of-the-art style transfer methods.

1. Introduction

Artistic text, or styled text, is a kind of art wildly used in design and media. As shown in Fig. 1, with text effects such as color, texture, shading, and extra decorative elements, artistic text becomes more visually pleasing and can vividly convey more semantic information. Traditionally, it needs complex manual operations to migrate text effects to other raw text, which is time-consuming especially when a bunch of text is to be processed. In this work, we propose a novel framework for transferring given text effects to arbitrary glyphs.

Text style transfer is a sub-topic of image style transfer. Although the task of image style transfer [8, 4, 12, 28, 6] have been wildly studied for years, text style transfer was not explored until recently. Yang *et al*. [23, 24] first explored this problem and designed a patch-based text effect transfer model. Due to the neglect of many important attributes such as decorative elements, directions, regular structures, *etc.*, it fails on many kinds of text styles. On the other hand, Azadi *et al*. [1] proposed a deep-based model, which is able to stylize capital English alphabets given a few shots. However, it can only generate images with a limited resolution of 64×64 and is hard to be applied to texts other than the 26 alphabets. Moreover, all these methods have assumed that the styles are uniform within or outside the text. Thus, exquisite decorative elements, which are commonly used in artistic text design, are ignored. These decorations are usually drastically different from the basal text effects and can make the text more visually impressive and more information expressed. Treating decorative elements and basal text effects as a whole style will seriously degrade the visual quality of the stylization results, as shown in Fig. 1.

To address this problem, in this paper, we propose a novel framework for text style transfer and pay special attention to decorative elements. The key idea is to detect, separate and recombine these important embellishments.

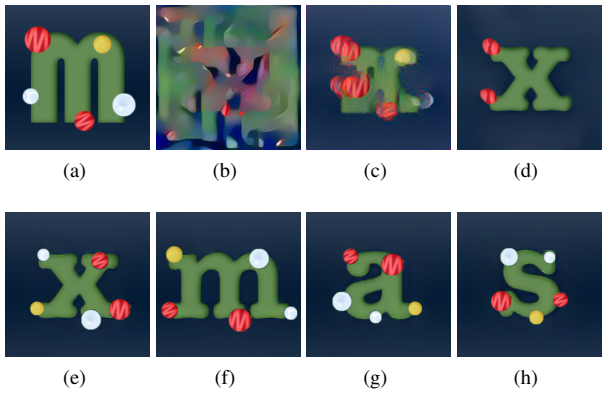


Figure 1: Demonstration of our text effects transfer method. (a) Input . (b) Neural Style Transfer [8]. (c) Neural Doodles [4]. (d) T-Effect [23]. (e)-(h) Our results, where both the basal text effects and the decorative elements can be transferred to the target text.

*Corresponding author.

This work was supported by National Natural Science Foundation of China under contract No. 61772043 and Beijing Natural Science Foundation under contract No. L182002 and No. 4192025.

First, we train a segmentation network to detect the decorative elements in the styled text. For training our segmentation network, we use synthetic data and further propose a domain-adaptation scheme so that the framework works well on real data. Then, based on the segmentation results, we are able to separate the decorative elements from basal text effects and design a text style transfer network to infer the basal text effects for the target text. To adapt our network to arbitrary text effects, a novel one-shot fine-tuning scheme is proposed, which empowers our network to extend to a new style with only one example required. Finally, cues for spatial distributions and element diversities are carefully characterized to jointly determine the layout of the decorative elements, which are then adaptively integrated onto the target text. Furthermore, to train the above models, we build a new dataset containing 59k professionally-designed styled texts with various text effects and fonts, and collect four thousand decorative elements and one thousand in-the-wild artistic texts from the web.

In summary, the contributions of this work are threefold:

- We define a new problem of text style transfer with decorative elements, and propose a novel framework to solve the problem. The scheme of separation and recombination of the basal text effects and the decorative elements empowers our method to adapt to different styles and glyphs.

- We train networks for effective decor detection and text effects transfer. Two corresponding novel training strategies are proposed to make the networks robust to arbitrary text styles. We propose a structure-aware decor recomposition method to determine the decor layout, which produces professional artistic typography.

- We introduce a new dataset containing thousands of styled text and decorative elements to support the training of our model.

2. Related Works

Image-to-Image Translation. The task of image-to-image translation is to translate an image from one domain into another, such as sketch to portrait [5], image colorization [26, 27], and rain removal [25, 18]. Hertzmann *et al*. proposed a nonparametric framework for single image pairs [11]. These years, benefiting from CNNs, data-driven methods have achieved great performance on many computer vision tasks. Combining Generative Adversarial Nets (GANs) [9], Isola *et al*. developed a common framework Pix2Pix [12]. This method is driven by paired data, which is sometimes hard to obtain. To get rid of this limitation, Zhu *et al*. designed CycleGAN [28] which can learn to translate images without paired ground truth. When facing N-domain translation problem, traditional models have to divide domains into pairs and be rebuilt $N(N-1)/2$ times. Choi *et al*. proposed to handle multi-domain translation with a single model StarGAN [6]. Although many researches have been done on image-to-image translation, few are targeted at styled text. Comprehensively considering the structure and the spatial distribution of artistic text, we propose a framework for transferring text effects.

Artistic Text Synthesis. Many researches have been conducted on font synthesis [20, 3, 17, 22]. However, generating text with artistic styles has not been widely studied. Most artistic text in daily life is carefully designed and produced by experts, and is hard to expand and migrate. Yang *et al*. [23] first proposed a texture-synthesis-based non-parametric methods for transferring text effects. However, this method needs careful parameter selections and fails on text effects with obvious structures, such as 'Wooden' and 'Stripes'. Azadi *et al*. [1] designed data-driven MC-GAN that can generate styled texts given a few examples and proposed a dataset containing 20k randomly synthesized color fonts and collected 910 styled texts from the internet. However, MC-GAN can only generate 26 English capital letters with a limited resolution of 64×64. Besides the low resolution, the synthetic color fonts in their dataset are quite different from artistic text used in common life. This dataset is not capable of training a network to produce high-resolution artistic text of various kinds. Moreover, decorative elements are quite common in styled text. However, they have never been considered in the aforementioned methods. We introduce a high-resolution dataset containing 59k artistic text, and proposed a framework which is able to create artistic typography with exquisite decor.

3. Style Transfer for Typography with Decor

The proposed text style transfer framework is shown in Fig. 2. In this paper, we focus on artistic text with two hybrid styles. For clarity, we define the decorative elements such as the red bowknot in Fig. 2 as **decor**. The remaining basal style excluding decor is referred to as **text effect**. We first extract a segmentation mask for decorative elements, where a domain adaptation strategy is applied for the robustness of the model on unseen styles (Sec. 3.1). Next, we transfer the text effects to the target text with decorative elements eliminated, during which we further propose an one-shot training strategy for improving the performance on unseen styles (Sec. 3.2). Finally, we recompose the artistic text and the decorative elements based on both the structure of the text and the spatial distribution of the decorative elements (Sec. 3.3).

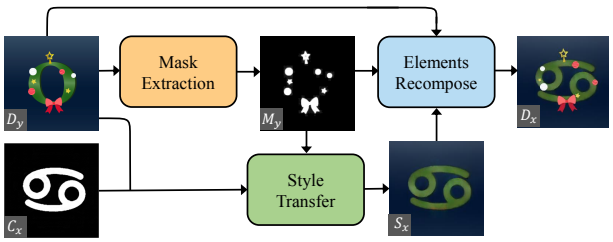
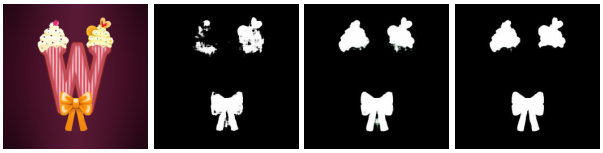


Figure 2: The proposed text style transfer framework. First, decorative elements are separated from the styled text. Then, text effects are transferred to the target text. Finally, the elements and the styled text are recomposed based on both the structure of the text and the spatial distribution of the decorative elements.



(a) (b) (c) (d)

Figure 3: Effect of the perceptual loss and domain adaptation. (a) Input. (b) Result with only L1 loss. (c) Result without domain adaptation. (d) Result with full loss.

3.1. Decorative Element Segmentation

We propose a segmentation network for decorative element detection. The network is trained on synthetic data, which will be introduced in Sec. 4. To reduce the gap between training data and real styled texts, we apply a domain adaptation strategy.

Segmentation Network. We adopt U-Net as the basic architecture of our segmentation network **netSeg**. As shown in Fig. 4, given the input artistic text D, the corresponding raw text C, the segmentation ground truth M and the prediction $\hat{M} = \mathbf{netSeg}(D, C)$, our network is tasked to approach the ground truth M in both L1 and perceptual senses. Thus the objective of **netSeg** can be expressed as

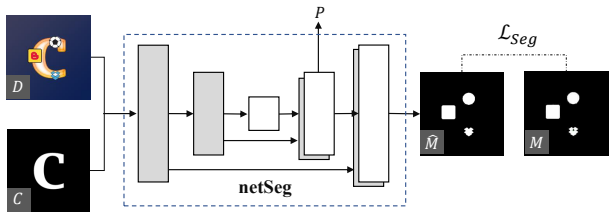
$$\mathcal{L}_{seg} = \lambda_{L1}\mathcal{L}_{L1} + \lambda_{Per}\mathcal{L}_{Per}, \tag{1}$$

where

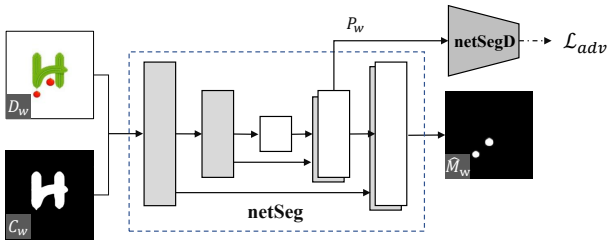
$$\mathcal{L}_{L1} = ||\hat{M} - M||_1, \tag{2}$$

$$\mathcal{L}_{Per} = ||\text{VGG}(\hat{M}) - \text{VGG}(M)||_1. \tag{3}$$

As illustrated in Fig. 3, the perceptual loss [13] helps the network better perceive the structure of the decor.



(a) Training **netSeg** in the source domain (synthetic styled text)



(b) Training **netSeg** in the target domain (real styled text)

Figure 4: The framework of the segmentation network with domain adaptation strategy. Discriminator is trained to distinguish the feature maps of the target from that of the source. The generator needs to fool the discriminator while giving segmentation predictions.

Adversarial-Loss-Based Domain Adaptation. There is a gap between synthetic data and real data in terms of color, decorative elements distribution, etc. Therefore, the network only trained on synthetic data can not well adapt to real styled texts, as shown in Fig. 3c. To address this issue, we apply a domain adaptation strategy for making the network more robust to the styled text in the wild.

The proposed domain adaptation strategy is similar to [21]. Here, the source domain is the synthetic training data, and the target domain is the real styled text. As shown in Fig. 4, in the phase of discriminator **netSegD**, it is trained to distinguish the feature map P of the second last layer of the generator. We exploit a cross-entropy loss for the discriminator:

$$\begin{aligned}\mathcal{L}_d(P) = -((1-z)\log(\mathbf{netSegD}(P)) \\ +z\log(\mathbf{netSegD}(P))),\end{aligned} \tag{4}$$

where $z = 0$ if the sample is drawn from the target domain, and $z = 1$ for the sample from the source domain. In the phase of generator, on the source domain the generator learns how to make segmentation predictions, while on the target domain it needs to fool the discriminator and reduce the gap between the two domains. The objective of the generator can be expressed as

$$\mathcal{L} = \lambda_{seg}\mathcal{L}_{seg} + \lambda_{adv}\mathcal{L}_{adv}, \tag{5}$$

where

$$\mathcal{L}_{adv} = -\log(\mathbf{netSegD}(P_w)) \tag{6}$$

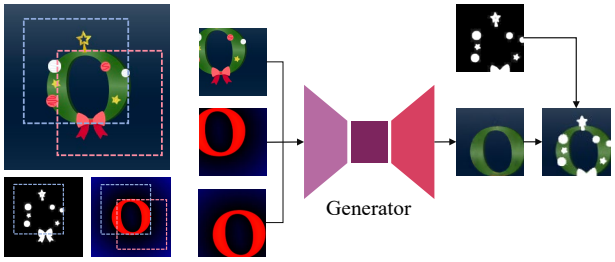


Figure 5: The one-shot training scheme.

is the adversarial loss for making the target feature map P_w closer to the source feature map P. As can be seen in Fig. 3d, the proposed domain adaptation can effectively improve the segmentation results.

3.2. Text Effect Transfer

Following the network architecture of Pix2pix [12], our text effect transfer model is a combination of U-Net [19] and PatchGAN [12]. Given D_y a styled text image with extra decorative elements, C_y the corresponding raw text image of D_y, and C_x a target raw text, the generator $\mathbf{G}$ learns to generate a fake styled text $S_x = \mathbf{G}(D_y, C_y, C_x)$, which has the text effects of D_y and the glyph of C_x. The discriminator $\mathbf{D}$ needs to distinguish whether the input is real or generated and whether it matches D_y, C_y and C_x or not. The loss function is a combination of WGAN-GP [10] and L1 Loss:

$$\mathcal{L}_{\mathcal{G}} = \lambda_{adv}\mathcal{L}_{adv} + \lambda_{L1}\mathcal{L}_{L1}, \tag{7}$$

where

$$\mathcal{L}_{L1} = ||S_x - \tilde{S}x||, \tag{8}$$

$$\begin{aligned}\mathcal{L}_{adv} = {} & \mathbb{E}_{\tilde{S}_x}[\mathbf{D}(\tilde{S}_x, D_y, C_y, C_x)] \\ & - \mathbb{E}_{S_x}[\mathbf{D}(S_x, D_y, C_y, C_x)] \\ & + \lambda_{GP}\mathbb{E}_{\hat{S}_x}[(||\nabla\mathbf{D}(\hat{S}_x, D_y, C_y, C_x)||_2 - 1)^2]],\end{aligned} \tag{9}$$

where $\tilde{S}_x$ is the ground truth, and $\hat{S}_x$ is uniformly sampled along the straight lines between the sampling of S_x and $\tilde{S}_x$.

One-shot Training Scheme. Learning-based image transfer methods often fail to perform well on unseen data. Moreover, it is impossible to collect a dataset covering all text effects that users may customize. As shown in Fig. 6b, through the aforementioned training strategy, our network learns to eliminate decorative elements and can generate the basic structure of the unseen text effects. But the unseen details cannot be properly reconstructed. To address this problem, we propose an one-shot fine-tuning scheme for unseen styles, where only one training pair is required.

(a) (b) (c)

Figure 6: Effect of our one-shot training scheme. (a) Input. (b) Result without one-shot fine-tuning. (c) Result after fine-tuning.

Specifically, we collect a bunch of patches randomly cropped from the styled text. They constitute a training set for the fine-tuning. The masks of the decors are then generated by the proposed segmentation network (Sec. 3.1). Using the segmentation mask, as illustrated in Fig. 5, we reduce the impact of decorative elements by not computing the L1 loss on these areas and blocking them before sending images to the discriminator. It is worth noting that unlike the pretraining process, during our one-shot fine-tuning, the ground truth decor-free image is not required. Out network can learn to both restore style details and eliminate the decor, which provides users with much more flexibility. As illustrated in Fig. 6c, with one-shot training the network can generate the iron edge and the red fabric texture.

3.3. Structure-Based Decor Recomposition

In this section, we propose to combine decorative elements and styled texts according to the structure of the styled text. First, we generate guidance maps characterizing the structure of the artistic text. Then we divide decorative elements into two classes based on their importance, and treat each class with a different transformation strategy. Finally, the elements and the styled text are combined to generate the final output.

Guidance Maps. We design four guidance maps characterizing the properties of the artistic text. These maps play important roles in the subsequent transformation.

- **Horizon Map.** Horizon map M_{Hor} identifies the position of pixels to the text in horizontal direction. Since human eyes are sensitive to edges, we amplify the horizontal changes near the edge of the text. We first define the gradient of M_{Hor} as G_{Hor}, which is initialized to one everywhere. Then G_{Hor} is adjusted according to the horizontal length of the text. For each y, define $x_{y,min}$ the leftmost point of the raw text on row y, and

$x_{y,max}$ the rightmost point, we generate $\tilde{G}_{Hor}$ by:

$$\tilde{G}_{Hor}(x,y) = G_{Hor}(x,y) + \begin{cases} (\mathcal{K}_w - |x - x_{y,min}|) * \mathcal{K}_s, & |x - x_{y,min}| < \mathcal{K}_w \\ (\mathcal{K}_w - |x - x_{y,max}|) * \mathcal{K}_s, & |x - x_{y,max}| < \mathcal{K}_w \\ 0, & else \end{cases} \tag{10}$$

where $\mathcal{K}_w = \mathcal{K}_{ws}(x_{y,max} - x_{y,min})$, $\mathcal{K}_{ws} < 0.5$. Here, $\mathcal{K}_{ws}$ and $\mathcal{K}_s$ control the width and the scale of the adjustment respectively. If row y has no overlap with the text body, we directly make $\tilde{G}_{Hor}(x,y) = G_{Hor}(x,y), \forall x$. With $\tilde{G}_{Hor}$, we build M_{Hor} by:

$$\begin{aligned} &M_{Hor}(x_{y,center}, y) = 0, \\ &M_{Hor}(x,y) - M_{Hor}(x-1,y) = \tilde{G}_{Hor}(x,y), \end{aligned} \tag{11}$$

where $x_{y,center}$ is the horizontal center of the text on row y. Finally, M_{Hor} is normalized to $[0,1]$ and slightly blurred to avoid drastic changes caused by complex edges. As illustrated in Fig. 7b, while representing the horizontal position of the text, changes near the edge of the text are amplified.

- **Vertical Map.** Vertical map M_{Ver} is similar to M_{Hor}, except that M_{Ver} identifies the property of the text in the vertical direction, as illustrated in Fig. 7c.
- **Distribution Map.** Distribution map M_{Dis} identifies the distance of pixels to the edge of the text, as illustrated in Fig. 7d. Given $Dis(x,y)$ the distribution-aware pre-prosessing map proposed in Sec. 4, M_{Dis} can be written as:

$$M_{Dis} = (1 - Dis(x,y))^{\mathcal{K}_{dis}}, \tag{12}$$

where $\mathcal{K}_{dis}$ controls the intensity of distribution map.

- **Existing Element Map.** Existing element map M_{Exi} is used to avoid overlapping. $M_{Exi}(x,y) = 0$ represents that no element has been placed on (x,y), and $M_{Exi}(x,y) = 1$ vice versa.

The final guidance map M_{guide} is a weighted concatenation of the previous maps:

$$M_{guide} = \text{Concat}[\lambda_i M_i | i \in \{Hor, Ver, Dis, Exi\}], \tag{13}$$

where Concat$[\cdot]$ indicates concatenation, and λ_i controls the weight of each map. The effect of each map is illustrated in Fig. 7. Without the horizontal or the vertical map, the elements will be crowded together. Without the distribution map, elements will depart from the text.

Decor Classification. We assume that there are two kinds of decorative elements: insignificant and significant ones.

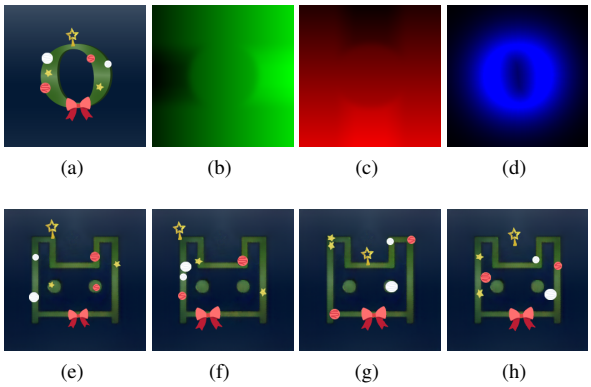


Figure 7: Guidance maps for the structure-based combination procedure. (a) Styled text with extra decorative elements. (b) The horizontal map. (c) The vertical map. (d) The distribution map. (e) Result with full map. (f) Result without the horizontal map. (g) Result without the vertical map. (h) Result without the distribution map.

The insignificant elements are repeatable, and are often randomly scattered on the text, for example, the colorful balls in Fig. 7a. The significant elements may play an important part in the semantic expression of the styled text, and are often single or paired, for example, the red bowknot in Fig. 7a.

Notice that insignificant elements usually share a similar shape with other elements, while significant elements often have a unique shape. For example, in Fig. 7a, the colorful balls share the same circle shape, while the shape of the red bow is not the same as other elements. Based on this observation, we classify elements via shape clustering. Then elements belong to size-1 clusters are regarded as significant, and the others are regarded as insignificant. Clustering can also divide insignificant elements into several groups, where elements in the same group share similar shapes. This clustering is implemented by first re-sizing the masks to 5×5, then clustering using DBSCAN [7].

Transformation and Combination. Given D_y an input artistic text with extra decorative elements, C_y the raw text of D_y, and C_x the target raw text, we can generate a segmentation mask M_y using the segmentation network proposed in Sec. 3.1, and the corresponding styled text (without decorative elements) S_x using the transformation network proposed in Sec. 3.2. Combining these, we finally transform decorative elements in consideration of both their distribution and the matching between them and the glyph.

We use DenseCRF [15] to refine the segmentation output and split different decorative elements by finding connected components. For each significant decorative element E on

(a) 60 text effects with 52 English letters

(b) Text effects from the web (c) Icons from the web

Figure 8: An overview of our styled text dataset. Our dataset consists of (a)-(c) three parts.

C_x, we find the area $E' \in C_y$ to place the element by:

$$\arg\min_{E' \in C_y} ||M_{guide}(E) - M_{guide}(E')||_2, \qquad (14)$$

where $M_{guide}(\cdot)$ indicates the average value of map M_{guide} for pixels on the specific area.

Then, elements are slightly re-sized and shifted to better fit the styled text. If the area where the element and the raw text overlap becomes smaller after placement, we zoom out the element and move it closer to the text, and vice versa.

For insignificant elements, we randomly exchange those elements in the same cluster and shift them by (14). This exchanging operation increases the variation of the result.

4. Data Collection and Augmentation

Styled Text Dataset. We introduce a new dataset including 60 different kinds of text effects with 52 English letters of 19 fonts, totally 59k images. We split 51k for training and the left for testing. Each styled text is of 320 × 320 and provided with its corresponding raw text. A few examples are illustrated in Fig. 8. We first collected Photoshop actions from the web or create actions following Photoshop tutorials. Then, we used batch processing in Photoshop to automatically replace characters and stylize raw texts. For decorative elements, we collected 4k icons from the web[1].

We also collected 1k artistic text of various text effects from the internet. For each, we generate a rough raw text by first thresholding then manually correcting some wrong parts. These styled texts in the wild are used for the domain adaptation of the segmentation network.

Training Data Generation and Pre-processing. Since carefully designed styled texts with extra decorative elements are hard to collect and annotate the mask of the decorative elements, we generate synthetic data for training through the following strategy. First, we randomly selected 0 to 5 decorative elements, and randomly placed them on the styled text while avoiding overlaps between the elements. The ground truth mask is generated using the alpha channel of the PNG-format decorative elements. Considering some elements are semi-translucent, instead of thresholding the alpha channels, we kept these semi-translucent masks.

[1] www.shareicon.net

(a) (b) (c)

Figure 9: Data pre-prosessing and augmentation. (a) Distribution-aware pre-prosessing. (b) Augmentation according to distribution. (c) Augmentation through texture combination.

In [23], Yang *et al*. pointed out that the patch patterns are highly related to their distance to the text skeleton. Take advantage of this feature, we pre-process the raw text images by calculating the distance of pixels to the text. This extra information is recorded on the B channel, while the R channel still keeps the original raw text, as illustrated in Fig. 9a. This pre-processing provides the network with distance distribution information.

Data Augmentation. Besides the 60 kinds of text effects, we design two types of augmentation. First, as shown in Fig. 9b, we generate random gradient colors according to the distances of pixels to the text, which strengthens the awareness of our network of distance distribution. Then, we collect 300 geometric and cartoon pattern images from the internet. While synthesizing, one of the pattern images is chosen as the background and another one is chosen to fill the text, as shown in Fig. 9c. This augmentation is used only for the segmentation network. The key idea is to increase the complication and variation of the patterns, so that the network can be more robust to various text effects.

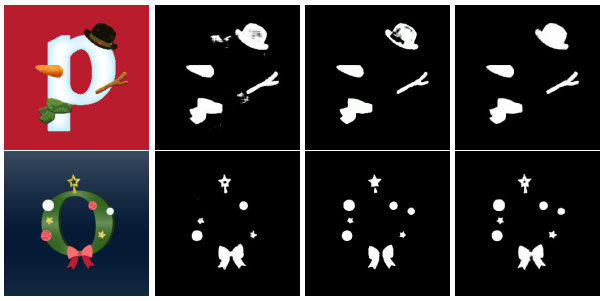
5. Experimental Results and Discussions

5.1. Implementation Details

For segmentation network, we use Adam with $\beta_1 = 0.5$, and $\beta_2 = 0.9$, and a learning rate 1e-4 with exponential decay strategy. The optimization is first performed with L1 loss for 30 epochs with a mini-batch size of 90, then performed with L1 loss and perceptual loss for 50 epochs with

Method	MAE	mIoU	Running Time
FCN8s	0.0155	0.8780	4.75s
U-Net	0.0040	0.953	6.10s
SegNet	0.0036	0.960	10.70s
Ours	0.0039	0.956	6.11s

Table 1: Quantitative comparisons of our segmentation network with traditional methods on the test set. We quantify the performance using mean absolute error (MAE), mean of intersection over union (mIoU), and total running time on the test set with GPU.



Input U-Net SegNet Proposed

Figure 10: Subjective comparisons of segmentation networks on real artistic text with extra decorative elements.

a mini-batch size of 15, and finally with full loss for 5000 iterations with a mini-batch size of 10. For style transfer network, a progressive growing strategy [14] is exploited to stabilize the training process. The resolution of the images increases from 64, 128, up to 256, and layers are gradually added to the front and rear of the generator. We also use Adam with $\beta_1 = 0.5$, and $\beta_2 = 0.9$, and a fixed learning rate of 2e-4. The mini-batch size is set to 200 at 64×64, 90 at 128×128 and 40 at 256×256, so that CPU and GPU resources can be fully utilized. The networks are trained on GTX 1080 GPU.

5.2. Decor Segmentation Comparative Results

Our segmentation network is compared with four classical semantic segmentation models, FCN8s [16], SegNet [2], and U-Net[2]. Among them, FCN8s and SegNet are based on classification models. These models are all trained on our training set with L1 loss, and tested on our test set. While testing, we still use randomly placed semantic decorative elements. This randomness may effect the performance, therefore we run the testing three times and calculate the average. By thresholding the ground truth and results into two class, we calculate the mean of intersection over union (mIoU).

From Table. 1, we can see that benefiting from the perceptual loss and domain adaptation, the performance rises compared to our U-Net baseline. Although the accuracy of our segmentation network is slightly lower than SegNet on test set, our segmentation network is more efficient and performs better on unseen text effects, which is clearly verified in Fig. 10 that the proposed segmentation network outperforms SegNet on real artistic texts with decorative elements. This is because the domain adaptation strategy helps the network to adapt to unfamiliar text effects and decorative elements.

5.3. Text Transfer Comparative Results

We first compare the proposed text effect transfer network with five state-of-art transfer methods in Fig. 11. The first two are image style transfer methods. Neural Style Transfer [8] uses CNNs to transfer the style of an image to another. It fails to find the correspondence between the style and the text. Therefore it twists the glyph and generates confused textures. Neural Doodles [4] uses neural-based patch fusion and has a context-sensitive manner in the algorithm. However it still twists the structure and the texture. The following two methods StarGAN [6] and Pix2Pix [12] are image-to-image translation methods based on GAN, and they are all re-trained on our dataset. StarGAN is a multi-domain translation model, whose domains in this task are the 60 different kinds of text effect. StarGAN is not capable of reconstructing details. The input of Pix2Pix [12] is revised to be the same as our input. Pix2pix generates wrong textures and artifacts. Benefiting from the progressive growing strategy and the WGAN-GP, our model is more stable than Pix2Pix. T-Effect [23] is a patch-based text effect transfer method. Although T-Effect is able to generate the main structure, it generates confused stripe textures. By comparison, our model is able to reconstruct vivid details, and fully adapt text effects to the given glyph. The ability of eliminating decor is further demonstrated in Fig. 12.

The framework is also compared with two one-shot methods Neural doodles and T-Effect on text effects not included in our dataset. Doodle still fails to well reconstruct the structure. Due to the neglect of regular structure and direction, T-effect distorts the texture, fails to preserve the background, and mistakes the shadow direction. Benefiting from the one-shot training scheme, our network is able to transfer the text effects even though these text effects are not included in the training set.

5.4. Comparative Results for Texts with Decor

We compare our full framework with T-Effect and Doodle on the task of transferring texts with decorative ele-

[2]We use PyTorch implementations. SegNet by https://github.com/zijundeng/pytorch-semantic-segmentation. FCN by https://github.com/meetshah1995/pytorch-semseg

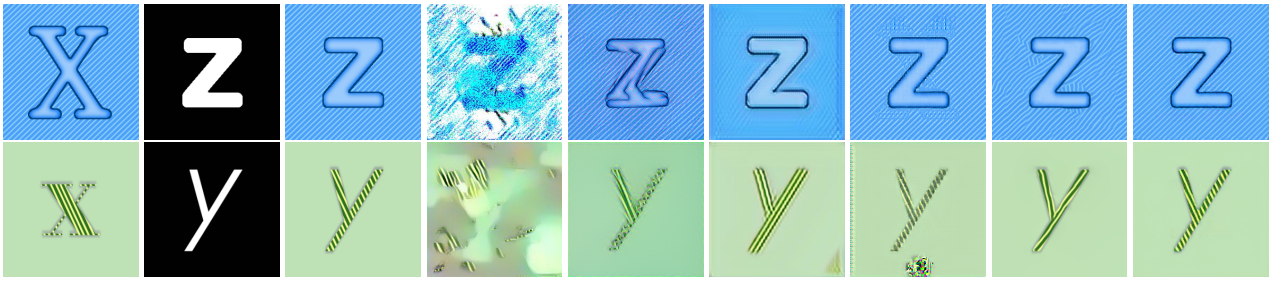


Figure 11: Subjective comparisons of text effect transfer.

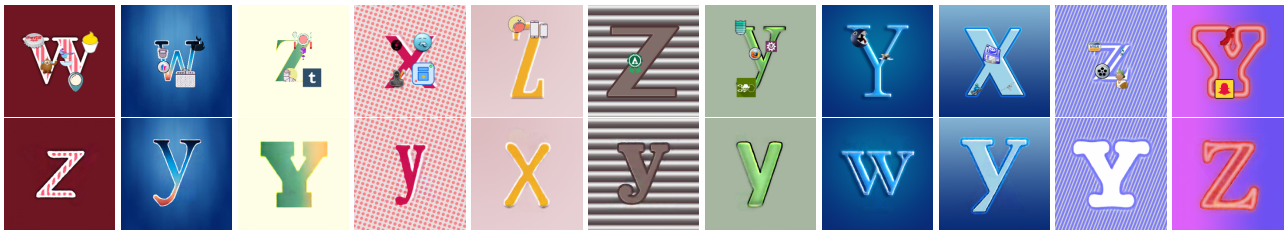


Figure 12: Subjective results of the proposed method for transferring basal text effects from the styled text with decor. First row: styled text with random decorative elements. Second row: our results.

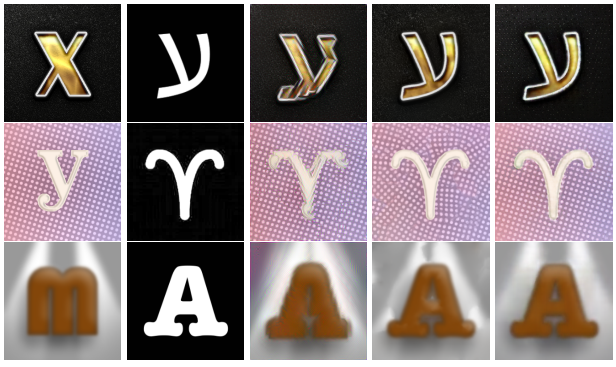


Figure 13: Subjective comparisons of text effects transfer for one-shot task.

Figure 14: Subjective comparisons of text effects transfer for one-shot task: (top-to-bottom) *Snowman*, *Xmas*, *Face*.

ments. Since neither T-Effect nor Doodle has a consideration of decorative elements, they all twist or drop the elements, therefore their results are not visually satisfying. Our framework not only successfully migrates the text effect onto the target glyph, but also preserves the intact form of the elements. Moreover, in our result, the elements are not just simply extracted and then placed. As shown in Fig. 14, in the *Snowman* artistic text, the black hat is resized to better fit the new glyph. In the *Xmas* artistic text, the little stars and balls are reshuffled, while the red bow is still at the bottom of the glyph.

6. Conclusion

In this paper, we propose a novel framework for transferring text effects with decorative elements. We first extract a segmentation mask for decorative elements, where a domain adaptation strategy is applied for improving the robustness. Next, we transfer the text effect to the target and eliminate decorative elements, during which we propose an one-shot training strategy for handling unseen styles. Finally, we recompose the artistic text and the decorative elements based on the structure of the text. Experimental results demonstrate the superiority of our framework.

References

[1] Samaneh Azadi, Matthew Fisher, Vladimir Kim, Zhaowen Wang, Eli Shechtman, and Trevor Darrell. Multi-content gan for few-shot font style transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 11, page 13, 2018. 1, 2

[2] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv preprint arXiv:1511.00561*, 2015. 7

[3] Neill DF Campbell and Jan Kautz. Learning a manifold of fonts. *ACM Transactions on Graphics (TOG)*, 33(4):91, 2014. 2

[4] Alex J Champandard. Semantic style transfer and turning two-bit doodles into fine artworks. *arXiv preprint arXiv:1603.01768*, 2016. 1, 7

[5] Tao Chen, Ming-Ming Cheng, Ping Tan, Ariel Shamir, and Shi-Min Hu. Sketch2photo: Internet image montage. In *ACM Transactions on Graphics (TOG)*, volume 28, page 124. ACM, 2009. 2

[6] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. *arXiv preprint*, 1711, 2017. 1, 2, 7

[7] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996. 5

[8] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2414–2423, 2016. 1, 7

[9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 2

[10] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, pages 5767–5777, 2017. 4

[11] Aaron Hertzmann, Charles E Jacobs, Nuria Oliver, Brian Curless, and David H Salesin. Image analogies. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 327–340. ACM, 2001. 2

[12] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *arXiv preprint*, 2017. 1, 2, 4, 7

[13] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, pages 694–711. Springer, 2016. 3

[14] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017. 7

[15] Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In *International Conference on Neural Information Processing Systems*, pages 109–117, 2011. 5

[16] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015. 7

[17] Huy Quoc Phan, Hongbo Fu, and Antoni B Chan. Flexyfont: Learning transferring rules for flexible typeface synthesis. In *Computer Graphics Forum*, volume 34, pages 245–256. Wiley Online Library, 2015. 2

[18] Rui Qian, Robby T Tan, Wenhan Yang, Jiajun Su, and Jiaying Liu. Attentive generative adversarial network for raindrop removal from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2482–2491, 2018. 2

[19] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 4

[20] Rapee Suveeranont and Takeo Igarashi. Example-based automatic font generation. In *International Symposium on Smart Graphics*, pages 127–138. Springer, 2010. 2

[21] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, and Manmohan Chandraker. Learning to adapt structured output space for semantic segmentation. *arXiv preprint arXiv:1802.10349*, 2018. 3

[22] Paul Upchurch, Noah Snavely, and Kavita Bala. From a to z: supervised transfer of style and content using deep neural network generators. *arXiv preprint arXiv:1603.02003*, 2016. 2

[23] Shuai Yang, Jiaying Liu, Zhouhui Lian, and Zongming Guo. Awesome typography: Statistics-based text effects transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7464–7473, 2017. 1, 2, 6, 7

[24] Shuai Yang, Jiaying Liu, Wenhan Yang, and Zongming Guo. Context-aware unsupervised text stylization. In *2018 ACM Multimedia Conference on Multimedia Conference*, pages 1688–1696. ACM, 2018. 1

[25] Wenhan Yang, Robby T Tan, Jiashi Feng, Jiaying Liu, Zongming Guo, and Shuicheng Yan. Deep joint rain detection and removal from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1357–1366, 2017. 2

[26] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European Conference on Computer Vision*, pages 649–666. Springer, 2016. 2

[27] Richard Zhang, Jun-Yan Zhu, Phillip Isola, Xinyang Geng, Angela S Lin, Tianhe Yu, and Alexei A Efros. Real-time user-guided image colorization with learned deep priors. *arXiv preprint arXiv:1705.02999*, 2017. 2

[28] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint*, 2017. 1, 2