FET-GAN: Font and Effect Transfer via K-shot Adaptive Instance Normalization

Wei Li,[1] **Yongxing He,**[1] **Yanwei Qi,**[1] **Zejian Li,**[1] **Yongchuan Tang**[1,2*]
[1]College of Computer Science, Zhejiang University, Hangzhou, 310027, China
[2]ZhejiangLab, Hangzhou, 310027, China
{liwei_2014, heyongxing, 21921305, zejianlee, yctang}@zju.edu.cn

Abstract

Text effect transfer aims at learning the mapping between text visual effects while maintaining the text content. While remarkably successful, existing methods have limited robustness in font transfer and weak generalization ability to unseen effects. To address these problems, we propose FET-GAN, a novel end-to-end framework to implement visual effects transfer with font variation among multiple text effects domains. Our model achieves remarkable results both on arbitrary effect transfer between texts and effect translation from text to graphic objects. By a few-shot fine-tuning strategy, FET-GAN can generalize the transfer of the pre-trained model to the new effect. Through extensive experimental validation and comparison, our model advances the state-of-the-art in the text effect transfer task. Besides, we have collected a font dataset including 100 fonts of more than 800 Chinese and English characters. Based on this dataset, we demonstrated the generalization ability of our model by the application that complements the font library automatically by few-shot samples. This application is significant in reducing the labor cost for the font designer.

1 Introduction

When designing a poster or web page, graphic designers adjust the visual effects of text or icons according to different themes. Such visual effects include colors, outlines, shadows, glows and textures. The consistent stylized effects of visual components allow the posters or web pages to be visually attractive. However, it is a tedious and repetitive work to apply stylized effects to all components manually. Similarly, when constructing a new font, the font designer also needs to style the glyph according to the specified theme. These style variations include factors like the stroke width, the appearance of serif, aspect ratio of the whole glyphs, and the curvature of lines. Different characters in one font library should have the same style, but it is tedious to manually apply these stylized effects to different characters. This problem is especially serious when the character set is very large, such as the Chinese or Japanese character set. In this work, we propose a general framework for synthesizing arbitrary stylized effects into target text with font variation with the help of deep neural networks. Furthermore, our model can extend the transfer from the text onto other graphic objects to keep the consistency of the style among different components.

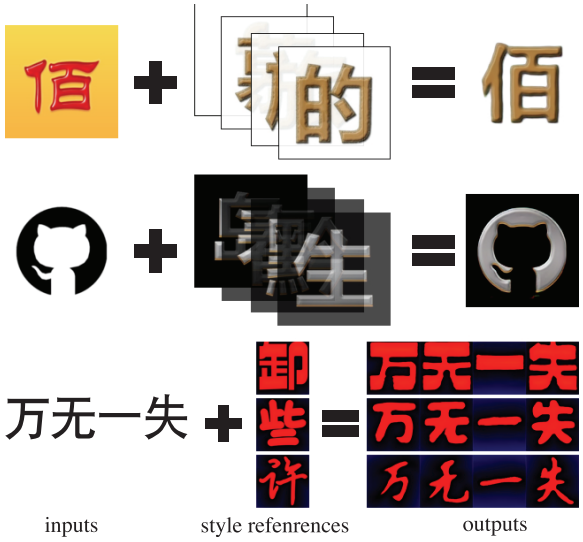


Figure 1: FET-GAN implements two functions. 1) It can transfer effects among texts in a few-shot manner. Particularly, the effects cover not only visual effects but also font variations. 2) It can translate the effects of text onto non-text graphic object. The font transfer demos in the last row are the foot stone of our font library complements application.

From the perspective of design aesthetics, the choice of fonts is highly correlated with visual expression. For example, the Times font is an appropriate choice for a formal academic poster, while the Comic Sans MS font can be used on the kindergarten website. If the two are exchanged, even if the other visual effects are unchanged, the overall style will be uncoordinated. Therefore, the typeface should be considered with the visual effect as a whole. However, in the existing research of text effect transfer, typeface and visual effect are treated as two isolated attributes (Yang et al. 2019; Yang et al. 2017). Among them, (Yang et al. 2019) requires to train multiple subnetworks to avoid interference from font

*Corresponding author

changes. Some networks are used to fetch the font image from image A, and the others transfer the effect from image B to the fetched font image. The output has to share the same typeface with A which is different from B. Besides, for different visual objects on the same page, such as text and icon, consistent stylization can make them harmonious and balanced. Unfortunately, there is no universal framework to transfer visual effects between text and graphic objects.

For humans, after changing the stylized effect of a text or graphic, we can still identify the original structure. For example, a triangle, whether changing color or adding texture, people can recognize that it is a triangle. Similarly, a letter W, with a different color, texture or typeface, one can still read it as W. From such observations, we assume that a visual object can be decomposed into global structural features and local effect features. The global structural features are stable and invariant to the change of effect features, and it always can be recognized correctly. Conversely, the local effect includes all other factors. Based on these assumptions, both typeface and visual effects can be classified as the local effect features.

As an attempt to separate the structure feature and the effect feature of an arbitrarily visual object, we propose the Font and Effect Transfer (FET-GAN) framework. It aims at learning a translation model. Given K samples in the same style for reference, the model can translate the original effects of an object to the referenced effect while maintaining the global structure features unchanged. Specifically, we use an encoder E to extract the implicit representation of the stylized effect as an effect code. As the K referenced samples have the same effect, the effect codes of them are encouraged to be the same. At the same time, we train a generator G with the adversarial training strategy (Goodfellow et al. 2014). Given the effect code, G translates the original effect of a visual object to the target effect indicated by the effect code. As a general framework, a trained model allows the stylized effect transfer between texts, as well as the translation of the effects from text onto the graphic objects as shown in Figure 1.

In summary, our contributions are fourfold:

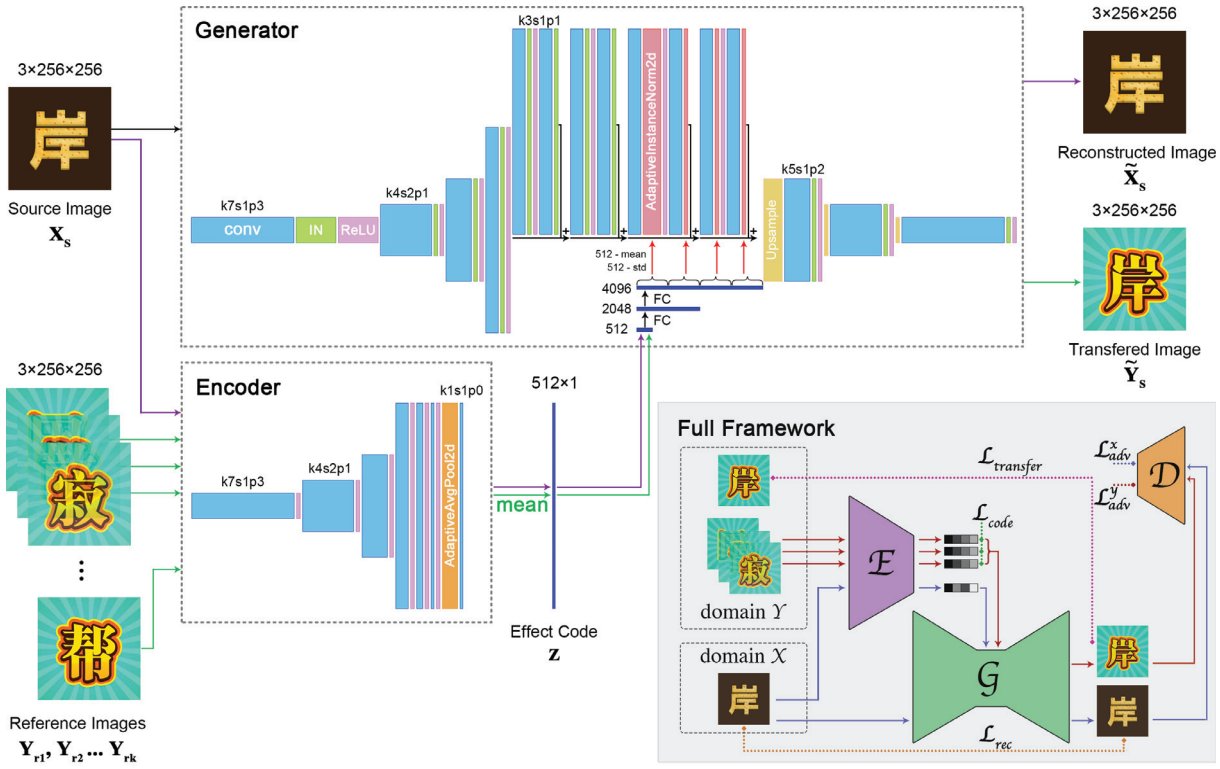
- We propose a new perspective which treats typeface as a part of the visual effect. Compared with the existing methods, the proposed approach can achieve effect transfer including typeface variation with a unified model.
- The proposed method learns the representation of effect in a few-shot learning manner and does not require a paired typeface-effect dataset. This representation approach can be generalized to non-text objects of any shape while reducing the difficulty of data collection.
- We come up with a few-shot finetune strategy to generalize our model to a new unseen effect.
- A new font dataset was collected on which we demonstrate the ability of our model to assist the font designer in complementing a font library. Given a few reference characters by font designers, the model can automatically complete other characters in the font library. This can significantly reduce the cost in a real-world application.

Our dataset, pre-trained models and codes in PyTorch (Paszke et al. 2017) are available at https://liweileev.github.io/FET-GAN/.

2 Related Work

Text effect transferring is related to the style transfer task. **Style Transfer** methods extract the style information from a reference image to synthesize an output based on the overall structure of a content image. In the seminal work of Gatys, Ecker, and Bethge (2016), the authors put forward that the style of an image is the information irrelevant to the position. They formulate the style representation by the second-order statistic of features in the form of the Gram matrix. And they refer to the high-level features as the content representation. All features are computed by the pre-trained VGG Network (Simonyan and Zisserman 2015). In the design of the objective function, the transferred image should match the style representation with the style image while matching the content representation with the content image. However, it relies on an optimization process, which is prohibitively slow. To speed up the stylization, (Li and Wand 2016; Johnson, Alahi, and Li 2016) attempt to train feed-forward networks that perform stylization with a single forward pass. But these methods suffer from a limitation that each network is only applicable to one or several specified styles. Significant efforts have been devoted to resolving this flexibility-speed dilemma such as (Huang and Belongie 2017; Li et al. 2017). To apply style transfer given an arbitrary style, they try to find a more general style representation as an alternative to the Gram Matrix. Among them, Huang and Belongie (2017) introduce an adaptive instance normalization layer to align the channel-wise mean and variance of the content features with those of the style features. This idea has been adopted to other tasks such as unsupervised translation (Liu et al. 2019) and high-quality generation (Karras, Laine, and Aila 2019). Inspired by this, we use a similar normalization method to introduce the target text effect as affine parameters in our effect transfer task.

Image-to-image Translation aims at learning an image generation function that maps an input image of the source domain to the target domain. Isola et al. (2017) propose the first unified translation framework in a supervised manner based on conditional GAN (Mirza and Osindero 2014). It combines an ℓ_1 loss and an adversarial loss with the paired data samples from two domains. To alleviate the problem of obtaining data pairs, unpaired image-to-image translation frameworks (Zhu et al. 2017; Liu, Breuel, and Kautz 2017) have been proposed. Among them, Zhu et al. (2017) introduce a cycle consistency loss to enforce the translated image could be translated back to the input one. This cycle consistency preserves key attributes between the input and the translated image. Liu, Breuel, and Kautz (2017) make a shared-latent space assumption that a pair of corresponding images in different domains can be mapped to the same latent representation in a shared-latent space. Our work is based on the shared latent space assumption but is designed for the few-shot arbitrary stylized effects transfer task. However, all these frameworks are only capable of learning the relations between two domains. To extend the translation ability in handling multiple domains, Choi et al. (2018) utilize a one-hot vector to specify the target domain. Nevertheless, the extension to a new domain is still expensive. To handle this, Liu et al. (2019) present a few-shot, unsuper-



(a) The architecture of encoder and generator.

(b) The framework of our model.

Figure 2: (a) The architecture of the encoder E and the generator G. E learns the effect code to express the stylized effect of reference images $y_1, \ldots, y_K$ or source image x. Then it feeds the effect code into G via adaptive instance normalization (Huang and Belongie 2017) after several fully-connected layers. G inputs an image x and outputs an image according to the effect code given by E. The output shares the same global structure with the input while keeping the effect from references. (b) The framework of FET-GAN with three subnetworks. The pictures are best viewed magnified on screen.

vised image-to-image translation algorithm that works on previously unseen target domain. Based on the above image-to-image translation methods, our work focuses on the text effects translation problem that maps an image of a source text effect domain to an arbitrary target text effect domain by a few-shot learning method.

Text Effects Transfer is first raised by Yang et al. in 2017. They match the stylized patches based on the normalized position between the strokes of glyphs. The method has appealing results but bears a heavy computational burden due to the optimal patch scale detection and distance estimation. Also, the structural differences of glyphs impact the transfer results a lot. TET-GAN (Yang et al. 2019) is the first to carry out text effects transfer via deep neural networks. With the help of conditional adversarial training, the authors try to disentangle glyph features from effect features by combining three encoders and two generators into three pairs of auto-encoder architectures. Except for the optimization burden of vast parameters in seven subnetworks (three encoders, two generators, and two discriminators), the model is also not robust to different glyph structure. Empirically, we find the generator interweaves the foreground and background in some cases. Additionally, all these frameworks only consider transferring visual effects without font variation. The transferred text still share the same font with the source one. In the work from Azadi et al. (2018), the authors try to handle this problem by combining font transfer and text effect transfer with two successive subnetworks. However, it can only handle 26 English capital letters with 64×64 resolution. In contrast, our approach uses an end-to-end framework to implement the text effects transfer with fonts variation. Apart from this, our model can be generalized to unseen text effects with a finetuning strategy. Further, our approach is stable enough to transfer effects to other non-text objects.

3 Method

In this part, we first describe FET-GAN, a framework to implement visual effects transfer with font variation among multiple text effects domains. Then, we will introduce how to use the few-shot finetune strategy to generalize the pre-trained effect transfer model to the unseen effect.

3.1 Framework

Our goal, as illustrated in Figure 2(a), is to train an encoder E and a generator G to learn mappings among multiple text effect domains. The encoder learns a latent representation from K reference samples in a few-shot manner. These K

reference samples belong to the same text effect domain Y. We refer to the learned latent representation as the effect code z. The generator is a network with a bottleneck similar to the auto-encoder architecture. G inputs a source image x_s from domain X and outputs an image $\tilde{y}_s$ conditioned on the effect code z. The output image $\tilde{y}_s$ shares the same structure with input image x_s. It also has the effect and typeface the same as the reference images. The effect code feeds into G via adaptive instance normalization after a few full connection operations. In order to make outputs diverse, we introduce the adversarial training (Goodfellow et al. 2014). To achieve our multi-domain transfer task, we generalize the patch-based discriminator proposed in (Isola et al. 2017) to multi-class version. The output of the original patch-based discriminator is a patch tensor whose channel number is 1. We change the number of channels in output to the number of classes of the training set as shown in Figure 3. Such a multi-class discriminator can be regarded as C discriminators that share parameters except for the last layer. C is the number of classes in the training set. Each class is a different stylized effect domain. For backpropagation, we only update the parameters in the channel whose class is corresponding to the reference images.

Figure 2(b) illustrates the whole training process of our proposed approach. Now, let's take a closer look at the design of the objective function.

Effect Code Consistency Loss. Here, we make an assumption that the effect codes for different samples in the same domain should be consistent. According to this assumption, we design the following constraint to the encoder:

$$L_{code} = \mathbb{E}_{y\sim Y} \sum_{i=1}^{K} \sum_{j=i}^{K} \|E(y_i) - E(y_j)\|_1 \tag{1}$$

This is inspired by (Kulkarni et al. 2015), which disentangles generative factors by pushing the latent codes of similar samples to be the same.

Transfer Loss. Unlike many other image-to-image translation tasks, there are corresponding samples among different domains for text effect transfer task. Since the total number of characters in one language is finite, it is achievable to obtain the same character in different text effects. For the paired samples of the same character in different domains, we propose the following transfer loss:

$$L_{transfer} = \mathbb{E}_{x\sim X, y\sim Y} \left\| y_x - G(x, \frac{\sum_{i=1}^{K} E(y_i)}{K}) \right\|_1 \tag{2}$$

where y_x is the target sample of x in domain Y. According to the assumption in loss (1), we take the mean of the effect codes of the K reference samples as the final effect code of domain Y.

This loss requires y_x, the target sample which gives a supervised signal. Without y_x, the training of this effect class is unsupervised. In the real-world situation, y_x may not be provided in every effect class, and the whole training is semi-supervised. In Section 4.4, we will compare the results of our model under supervised, unsupervised, and semi-supervised conditions by random eliminating the target samples.

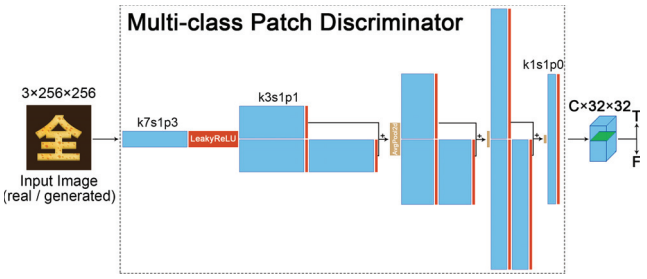


Figure 3: The architecture of discriminator D. The number of channels in output is the number of classes in the training set denoted as C. We calculate the loss by taking the channel corresponding to the class of the input.

Reconstruction Loss. This is inspired by the cycle consistency idea from (Zhu et al. 2017). If we feed the effect code of the source image x_s into G, the output image $\tilde{x}_s$ should be exactly the same as the input source:

$$L_{rec} = \mathbb{E}_{x\sim X} \|x - G(x, E(x))\|_1 \tag{3}$$

This loss is helpful to guarantee the effect code extracted by the encoder is effective. With the aid of transfer loss and reconstruction loss, the outputs of G vary by references although the inputs are the same.

Adversarial Loss. To make the generated images indistinguishable from the real ones, we adopt an adversarial loss of the hinge version (Miyato et al. 2018). On top of that, we introduce the gradient penalty regularization proposed in (Mescheder, Geiger, and Nowozin 2018) to the real samples to ensure the convergence and stability of training. For source inputs x and the reconstructed outputs $G(x, E(x))$, the adversarial loss is computed only using the corresponding prediction score of discriminator D in the channel C_x (denoted as D^{C_x}) as follows:

$$\begin{aligned} L_{adv}^{\mathcal{X}} = {} & \mathbb{E}_{x\sim X}[\min(0, -1 + D^{C_x}(x)] \\ & + \mathbb{E}_{x\sim X}[\min(0, -1 - D^{C_x}(G(x, E(x))))] \\ & + \lambda_{GP} \mathbb{E}_{x\sim X} \|\nabla D(x)\|^2 \end{aligned} \tag{4}$$

where λ_{GP} is the weight of the gradient penalty regularization term. We use $\lambda_{GP} = 10$ indicated by the original paper.

Similarly, for reference images y and the transferred outputs $G(x, E(y))$, the loss is computed with the prediction score of D^{C_y}:

$$\begin{aligned} L_{adv}^{\mathcal{Y}} = {} & \mathbb{E}_{y\sim Y}[\min(0, -1 + D^{C_y}(y)] \\ & + \mathop{\mathbb{E}}_{\substack{x\sim X, \\ y_1,\ldots,y_K\sim Y}} [\min(0, -1 - D^{C_y}(G(x, \frac{\sum_{i=1}^{K} E(y_i)}{K})))] \\ & + \lambda_{GP} \mathbb{E}_{y\sim Y} \|\nabla D(y)\|^2 \end{aligned} \tag{5}$$

Full Objective. Finally, the objective function is as follow:

$$\begin{aligned} L = {} & \lambda_{code} L_{code} + \lambda_{transfer} L_{transfer} + \lambda_{rec} L_{rec} \\ & + \lambda_{GAN} (L_{adv}^{\mathcal{X}} + L_{adv}^{\mathcal{Y}}) \end{aligned}$$

where λ_{code}, $\lambda_{transfer}$, λ_{rec} and λ_{GAN} are hyper-parameters that control the relative importance of the losses.

The components of our proposed model G, E and D are optimized via the following optimization problem:

$$\min_{G,E} \max_{D} L$$

3.2 Finetune Strategy for Generalization

Models trained by deep networks rely heavily on the distribution of training sets. For new samples that do not obey the training set distribution, the performance of the model will plummet. To extend our method to support the arbitrary effect transfer task, we come up with a finetune strategy based on the pre-trained model. This finetuning strategy only requires one or several samples from a new effect.

Given samples from the new effect, we randomly flip and crop them to make a temporary dataset. All images in the temporary dataset have the same effect but differ in structure or position. After building the dataset, we initialize a model based on the well-trained one and then finetune it. Because the samples in the current dataset do not belong to any class of the original datasets, in order to add new effect class, we append a new channel to the output of the discriminator. We take a random batch of real samples in the temporary dataset and send them to the pre-trained discriminator to obtain the predictions of all C channels. The maximum one from these predictions is taken and the parameters of this channel is used as the initialization of the $C+1$ channel. As explained before, the outputs of the discriminator in different channels indicate whether the sample comes from the domain corresponding to this channel. Such an initialization method can select the existing effect which is most similar to the new effect. Then, we finetune a new model to support the new effect.

Notice that the source image and the reference images come from the same effect domain in the finetuning process. This means the transferred image in loss (2) and the reconstructed image in loss (3) look the same. The two losses encourage the encoder to extract the effect feature form the temporary dataset. In Section 4.5, we will compare the transfer results before and after the finetuning on an unseen effect.

4 Experiments

4.1 Datasets

TextEffects Dataset: We use the text effects dataset proposed in (Yang et al. 2019). This dataset is paired that each text effect image is provided with its font image. Based on our assumption that font is a part of the visual effect, we separate 6 fonts and combine them with the original 64 effects. Finally, there are a total of 70 classes of text effects.

Fonts-100: We collect a new dataset including 100 fonts each with 775 Chinese characters, 52 English letters, and 10 Arabic numerals. There are a total of 83,700 images, each of which is 320×320 in size. Figure 4 shows an overview of these fonts for the same Chinese character. This Fonts-100 dataset is used to demonstrate the ability of our model to assist the font designer in complementing characters in font library. We take the first 80 fonts for training, and the remaining 20 as unseen for finetuning. Specific experimental configuration and results are in Section 4.6.

Figure 4: A demonstration of the Fonts-100 dataset. We choose this Chinese character (read as yong) because it is known for containing all the stroke types.

4.2 Comparison

We conduct five comparison experiments of our model with approaches for the font and effect transfer task as shown in Figure 5, each row being a group. These approaches are TET-GAN (Yang et al. 2019), T-Effect (Yang et al. 2017), CycleGAN (Zhu et al. 2017), StarGAN (Choi et al. 2018) and AdaIN (Huang and Belongie 2017).

For FET-GAN, we set $\lambda_{code} = 1$, $\lambda_{transfer} = 10$, $\lambda_{rec} = 10$ and $\lambda_{GAN} = 1$. We optimize the objective using Adam solver (Kingma and Ba 2014) with a batch size of 4. All networks are trained from scratch with a learning rate of 0.0002. Based on the TextEffects dataset with 70 effects, we train the FET-GAN model using $K = 4$ for 30 epochs and test it with $K = 1$.

In the first row, we compare the results of text effect transfer with font variation. For this task, we expect models should output the image which has the structure of the source and the visual effects and fonts of the reference. Specially, the glyph image of the reference is essential to construct the analogy pairs for T-Effect. We put them in the lower-left corner of the result. We can find that CycleGAN and AdaIN only capture partial color feature of the strokes and background. But the transfer of outline effect and typeface are failed. TET-GAN, T-Effect perform the visual effect transfer well, but can not transfer the typeface. Besides, the result of T-Effect is less natural and smooth than another three methods. StarGAN tries to capture the change of typeface, but not work well. The result is between the two typefaces of source and reference. In contrast, the transfer result of FET-GAN is closest to that of human.

The second row demonstrates the robustness comparison for source input. We believe that all text effects contain the same global structural information about the text. In this experiment, we change the source to a different style which does not exist in the training dataset to test the robustness of models for source input. We can find that TET-GAN, CycleGAN and AdaIN interwove the foreground and the background. The results of StarGAN and FET-GAN are similar to that of human while the output of FET-GAN is much clearer.

The third row shows the comparison of the effect transfer between complicated effects. The source and the reference are different in color, texture, outline, background, and typeface. T-Effect and AdaIN are unable to transfer effects, and even fail to obtain a complete structure. TET-GAN need to de-stylize the source to the font image, then stylize the output font image to the reference effect. CycleGAN, StarGAN, and FET-GAN can get appealing results. Among them, Cycle-

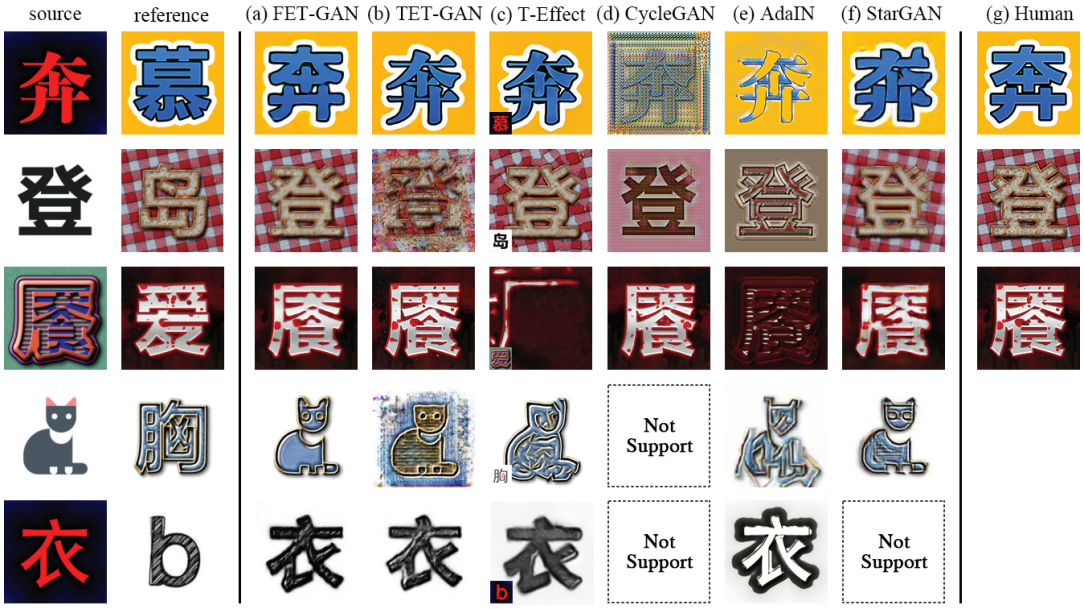


Figure 5: Comparisons with state-of-the-art methods on font and effect transfer task. Manual results by human are shown as ground truth. Five rows correspond to (1) text effect transfer with font variation, (2) robustness comparison for source, (3) complicated effect transfer, (4) effect translation from text to graphic, (5) generalization comparison of unseen effect, respectively.

GAN has the highest training cost and StarGAN is the least clear. In contrast, our model is both robust and effective.

The stylized effects translation from the text onto a non-text graphic object are shown in the fourth row. Since the source image is completely different from the domains in the training dataset, CycleGAN does not support such a transfer. For the remaining five models, only FET-GAN can get an appealing result. There are structural errors or texture disorder in all other results.

The last row is the generalization results of new effect not included in the dataset. As domain adaption models based on the training dataset, CycleGAN and StarGAN do not support this kind of transfer. Since this is an ill-posed problem, there is no ground-truth as a standard. Through observation, we find that the results of TET-GAN and FET-GAN are similar, the result of T-Effect is relatively blurry, and the results of AdaIN differ most from the reference.

4.3 The Influence of K

The influence of K needs to be discussed in three cases: training, testing and finetuning. We conduct the experiments under these three situation with different K as shown in Figure 6. All other hyper-parameter settings are the same with the Section 4.2.

In the training stage, we train four models with K=1,2,4,8, respectively. After training for 30 epochs, we test these models with the same source and reference input. And the number of reference input for test is 1. The result is in the first row of Figure 6. We can find that when $K = 1$ the transfer result is not satisfactory. As K increases, the effects of the transferred get closer to the reference.

For the test stage, we use the model trained with $K = 4$. With the same source, we input the different number of reference images from the same effect domain to compare the results. As illustrated in the second row, the number of reference images do not influence the result. This is in line with expectations. Equation (1) ensures effect codes of samples in the same domain to be identical, so the model is less sensitive to variations.

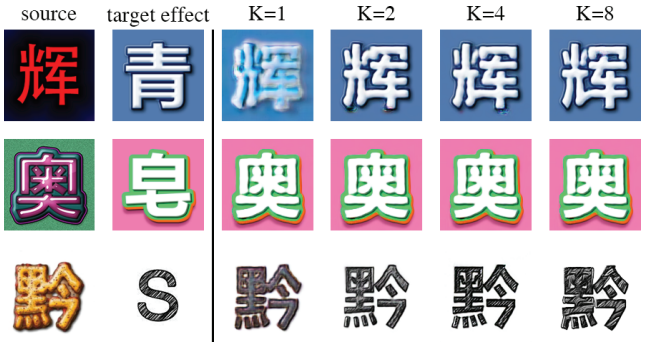


Figure 6: Comparisons under different K. Three rows correspond to training, test and finetuning, respectively.

The influence of K on the finetuning stage is similar to that of the training stage. After finetuning with different K, we test the models with the same source and reference input. We find that the larger the K is, the better the effect is.

4.4 Ablation Study

In Figure 7, we study the effect of the losses. We train all models with $K = 4$ for 30 epochs and compare the results with the same inputs. Without L_{code}, the transferred result

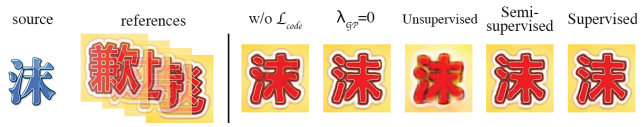
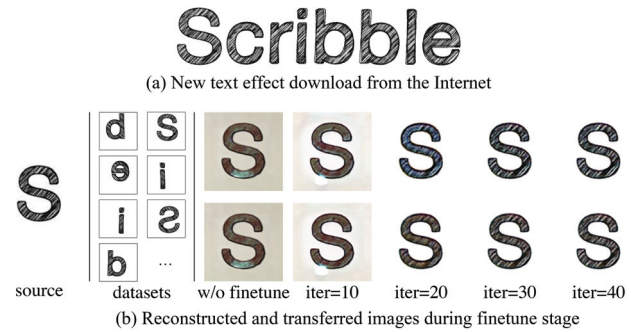


Figure 7: Effect of L_{code}, gradient penalty regularization and $L_{transfer}$. By eliminating some or all of the target samples, the task becomes semi-supervised or unsupervised.

looks unrealistic with uneven color and unclear outline. Gradient penalty regularization is removed by setting $\lambda_{GP} = 0$. We find that white or black spots appear due to the training instability of GAN. Without $L_{transfer}$, the training becomes unsupervised. After training for 30 epochs, the transferred image is still blurry. We also perform semi-supervised experiments by randomly removing 50% target samples. Compared with the supervised result, after 30 epochs training, the result of semi-supervised is similar, but the convergence is slower. From the perspective of data collection, semi-supervised is more in line with the real-world requirement.

4.5 Generalization



(a) New text effect download from the Internet

(b) Reconstructed and transferred images during finetune stage

Figure 8: The generalization to a new effect.

In this experiment, we verify the effectiveness of the finetuning strategy, which is used to extend the trained model to the new effect. We download a new text effect from the Internet as shown in Figure 8(a). As described in Section 3.2, a temporary dataset is built by random flip and crop the samples. Then, we finetune the model with 40 iterations and the intermediate results are shown in Figure 8(b). Since source and reference images belong to the same domain, reconstructed and transferred results are always the same. Unfortunately, we find that when the number of iterations is too large, the over-fitting phenomenon arises and the results are poor due to the simplicity of the dataset. Therefore, how to choose a reasonable number of iterations remains unsolved.

4.6 Application: Complement Font Library Automatically

We design an experiment of complementing characters in the font library automatically. In Figure 9, we show the results based on the 100-Fonts datasets. For training, we randomly

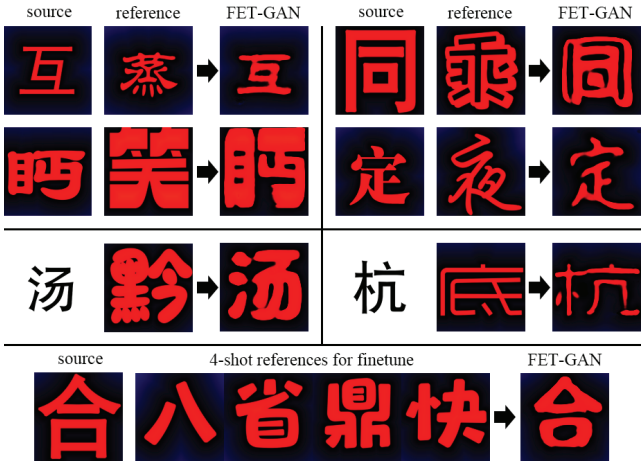


Figure 9: Using FET-GAN to complement characters in the font library. The first two rows are font transfer examples of characters in the dataset. The third row demonstrates the results of characters not in the dataset. The last row shows generalization result to a new font after 4-shot finetuning.

select 80 fonts as the training set and let $K = 4$ to train 30 epochs. In the first two rows, there are four transfered examples of characters in the dataset. The transferred images all retain the structure feature of the source and the typeface feature of the reference. In the third row, we pick two characters not included in the 837 characters. The results are also satisfactory. In the last row, we randomly choose a new font in the remaining 20 fonts of Fonts-100 and finetune the trained model by four samples for 40 iterations. After finetuning, our model can also perform font transfer task for the new font.

In summary, our model can implement the font transfer task. In particular, the latter two experiments exemplify that our proposed method is helpful in the automatic construction and completion of a font library. This application is significant in reducing labor cost for font designers.

5 Conclusion

In this paper, we propose FET-GAN, which implements text effect transfer with font variation among multiple text effects domains. This model can also achieve a stable result on effect translation from text to graphic objects. Besides, we come up with a new few-shot finetuning strategy to make the model generalizable to the new effect. Extensive experiments show that our approach outperforms dedicated approaches in the text effect transfer task. Furthermore, we collect a new dataset with 100 fonts and develop a character complement application in font libraries. Honestly speaking, both the network architectures and loss designs in our method are not new topics in the image-to-image translation community, but the character disintegration perspective in our model can significantly reduce the labor cost for text effect and font designers in practice. We hope our work would open the path for automatic font library completion, and the proposed few-shot finetuning strategy can be adapted to other training techniques.

Acknowledgments

This work is funded by Scientific Innovation 2030 - Project of Artificial Intelligence in the Next Generation (No. 2018AAA0100700), the National Natural Science Foundation of China (NO.61773336 and NO.91748127) and Scientific Plan Project of Zhejiang Province (No. 2019C03137). We would like to extend our gratitude to data providers.

References

Azadi, S.; Fisher, M.; Kim, V.; Wang, Z.; Shechtman, E.; and Darrell, T. 2018. Multi-content GAN for few-shot font style transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 11, 13. IEEE.

Choi, Y.; Choi, M.; Kim, M.; Ha, J.-W.; Kim, S.; and Choo, J. 2018. StarGAN: Unified generative adversarial networks for multi-domain image-to-image translation. In Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; and Garnett, R., eds., *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 8789–8797. IEEE.

Gatys, L. A.; Ecker, A. S.; and Bethge, M. 2016. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2414–2423. IEEE.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. In Ghahramani, Z.; Welling, M.; Cortes, C.; Lawrence, N. D.; and Weinberger, K. Q., eds., *Advances in neural information processing systems (NeurIPS)*, 2672–2680. Curran Associates, Inc.

Huang, X., and Belongie, S. 2017. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision (CVPR)*, 1501–1510. IEEE.

Isola, P.; Zhu, J.-Y.; Zhou, T.; and Efros, A. A. 2017. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 1125–1134. IEEE.

Johnson, J.; Alahi, A.; and Li, F. F. 2016. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision (ECCV)*, 694–711. Springer.

Karras, T.; Laine, S.; and Aila, T. 2019. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4401–4410. IEEE.

Kingma, D., and Ba, J. 2014. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*.

Kulkarni, T. D.; Whitney, W. F.; Kohli, P.; and Tenenbaum, J. 2015. Deep convolutional inverse graphics network. In Cortes, C.; Lawrence, N. D.; Lee, D. D.; Sugiyama, M.; and Garnett, R., eds., *Advances in neural information processing systems (NeurIPS)*, 2539–2547. Curran Associates, Inc.

Li, C., and Wand, M. 2016. Precomputed real-time texture synthesis with markovian generative adversarial networks. In *European Conference on Computer Vision (ECCV)*, 702–716. Springer.

Li, Y.; Fang, C.; Yang, J.; Wang, Z.; Lu, X.; and Yang, M.-H. 2017. Universal style transfer via feature transforms. In Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; and Garnett, R., eds., *Advances in neural information processing systems (NeurIPS)*, 386–396. Curran Associates, Inc.

Liu, M.-Y.; Huang, X.; Mallya, A.; Karras, T.; Aila, T.; Lehtinen, J.; and Kautz, J. 2019. Few-shot unsupervised image-to-image translation. In *The IEEE International Conference on Computer Vision (ICCV)*.

Liu, M.-Y.; Breuel, T.; and Kautz, J. 2017. Unsupervised image-to-image translation networks. In *Advances in neural information processing systems (NeurIPS)*, 700–708. Curran Associates, Inc.

Mescheder, L.; Geiger, A.; and Nowozin, S. 2018. Which training methods for GANs do actually converge? In Dy, J., and Krause, A., eds., *Proceedings of the 35th International Conference on Machine Learning (ICML)*, volume 80 of *Proceedings of Machine Learning Research*, 3481–3490. Stockholmsmssan, Stockholm Sweden: PMLR.

Mirza, M., and Osindero, S. 2014. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*.

Miyato, T.; Kataoka, T.; Koyama, M.; and Yoshida, Y. 2018. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations (ICLR)*.

Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; and Lerer, A. 2017. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*.

Simonyan, K., and Zisserman, A. 2015. Very deep convolutional networks for large-scale image recognition. *International Conference on Learning Representations (ICLR)*.

Yang, S.; Liu, J.; Lian, Z.; and Guo, Z. 2017. Awesome typography: Statistics-based text effects transfer. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2886–2895.

Yang, S.; Liu, J.; Wang, W.; and Guo, Z. 2019. TET-GAN: Text effects transfer via stylization and destylization. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 33, 1238–1245.

Zhu, J.-Y.; Park, T.; Isola, P.; and Efros, A. A. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2223–2232. IEEE.