TextPainter: Multimodal Text Image Generation with Visual-harmony and Text-comprehension for Poster Design

Yifan Gao*
University of Science and Technology
of China
Hefei, China
eafn@mail.ustc.edu.cn

Jinpeng Lin
Alibaba Group
Beijing, China
linjinpeng.ljp@alibaba-inc.com

Min Zhou
Alibaba Group
Beijing, China
yunqi.zm@alibaba-inc.com

Chuanbin Liu†
University of Science and Technology
of China
Hefei, China
liucb92@ustc.edu.cn

Hongtao Xie
University of Science and Technology
of China
Hefei, China
htxie@ustc.edu.cn

Tiezheng Ge
Yuning Jiang
Alibaba Group
Beijing, China
tiezheng.gtz@alibaba-inc.com
mengzhu.jyn@alibaba-inc.com

ABSTRACT

Text design is one of the most critical procedures in poster design, as it relies heavily on the creativity and expertise of humans to design text images considering the visual harmony and text-semantic. This study introduces TextPainter, a novel multimodal approach that leverages contextual visual information and corresponding text semantics to generate text images. Specifically, TextPainter takes the global-local background image as a hint of style and guides the text image generation with visual harmony. Furthermore, we leverage the language model and introduce a text comprehension module to achieve both sentence-level and word-level style variations. Besides, we construct the PosterT80K dataset, consisting of about 80K posters annotated with sentence-level bounding boxes and text contents. We hope this dataset will pave the way for further research on multimodal text image generation. Extensive quantitative and qualitative experiments demonstrate that TextPainter can generate visually-and-semantically-harmonious text images for posters.

CCS CONCEPTS

• **Computing methodologies** → **Computer vision**; *Natural language processing*; • **Applied computing** → Arts and humanities.

KEYWORDS

text image generation, poster design, text comprehension

---

*This work was done when Yifan Gao was at Alibaba as an intern.
†Corresponding author.

---

1 INTRODUCTION

Text design is an important subtask in poster design, where people create and render harmonious text for posters to convey information clearly and effectively. With the development of deep learning, researchers have made a series of successful attempt around poster design [10, 28], e.g. layout generation [3, 38], text content generation [5]. However, there has been little research conducted on text design.

In this paper, we propose a novel task entitled Text Image Generation for Posters (TIGER), which aims to accomplish text design with the automated method. Specifically, the goal is to generate text images for a specific line based on its position and text content. As demonstrated in Figure 1, generating high-quality text images which are clear, harmonious in color, and semantically appealing is a challenging task. Though this task seems to be similar to font generation [7, 24, 36] or scene text generation tasks [20, 32, 35], it has three main unique and challenging features.

Firstly, the generation style should not be explicitly predefined but implicitly relevant to the background. The goal of TIGER is to create a visually pleasing text image by skillfully integrating the text content with the poster background in a natural, beautiful, and attractive manner. We aim to produce a distinctive visual effect that is not achieved by imitation of a predefined style. However, the font or scene text generation is to extract the predefined style as a reference and transfers it to other text images. For instance, the text colors in the first and second columns of Figure 1 are consistent with their corresponding subjects and clearly distinguishable from the surrounding background regions.

Secondly, during the rendering of the generated text images, each character should be differently designed according to its semantics.

Figure 1: Text image generation for poster design using TextPainter (English translation in brackets). (Top) Given a clean image without filled text, the content and position of the text. (Bottom) TextPainter generated harmonized text images. TextPainter is sensitive to the content of poster background image (left & middle). TextPainter is capable of highlighting keywords (right).

Comprehending the poster copy can effectively emphasize the keywords that are meant to be exhibited to the audience. As shown in the last column of Figure 1, the highlighted keywords, are differentiated from others through distinct styles. In contrast, the font or scene text generation task usually demonstrates independent visual style and text semantics.

Finally, the present absence of a properly structured and annotated dataset for this task exacerbates the challenge of training models. In particular, textual style attributes such as font, color, opacity, and outline present difficulties in manual annotation. Besides, image size and proportions vary and are not consistent.

To mitigate the aforementioned challenges, we propose a novel multimodal approach named TextPainter that exploits the contextual visual style of posters and the corresponding text semantics to produce text images that are both visually and linguistically meaningful. Specifically, our approach initially approximates the overall color style of the text image by analyzing the global-local information of the poster background. Furthermore, it refines the fine-grained color of specific characters of the text image to match the text semantics at both the word and sentence levels, resulting in more innovative designs.

Firstly, we introduce a StyleGAN-based method for generating visually harmonious text images. Specificall, the text initially is rendered as an image, then encoded by the glyph encoder, which is used as input to the generator along with the background patch

of the text area. Simultaneously, a color style encoder is utilized to extract the implicit global and local styles of the poster background and guide the generation process.

Secondly, a text understanding module is introduced to enhance the visual design ability of TextPainter through the utilization of text semantics. Specifically, we use Language-Image Pre-training (CLIP) models to extract text semantic tokens and then fuse word-level and text-level tokens with visual features in various methods.

Finally, we employed image-level loss functions and several training techniques to facilitate the effective training of TextPainter. These approaches effectively address the issues of the dataset with weakly supervised annotations and variations in image size.

Besides, we constructed a dataset, called PosterT80K, to validate our proposed approach. It consists of 87,529 posters and 342,579 text elements collected from real-world use cases. Each element has been annotated with its bounding box and content. In particular, it was demonstrated through extensive experiments conducted on the dataset, which showed the effectiveness of our approach. Our main contributions can be summarized as follows:

- We have proposed a new task, text image generation for poster design, which is aimed to generate clear, color harmony and creative text images pixel by pixel on poster backgrounds.
- We propose the TextPainter that utilizes the contextual visual style of the poster and the corresponding text semantics for text image generation, which is the first method to utilize text semantics to help text image generation to the best of our knowledge.
- We construct a large-scale poster dataset, Poster-T80K, consisting of 87,529 posters designed by designers with sentence-level bounding boxes and content of text string annotations.

2 RELATED WORK

2.1 Image-to-image Generation and Style Transfer

With the advance of generative adversarial networks [9], the quality of image generation is getting improved and the controllability of the generated content is getting easier. Pix2pix [16] proposed a general solution to image-to-image generation problems using conditional adversarial networks. Wang et al. [30] extended these efforts to high-resolution image generation. Style transfer can be seen as a special kind of image-to-image translation task, which modifies attributes of images, such as their style while keeping their content. Gatys et al. [8] published the first neural algorithm that creates artistic images of high perceptual quality by a pre-trained convolutional neural network. Using adaptive instance normalization (AdaIN) that affines transformation parameters in normalization layers to represent styles. Huang et al. [13] propose a method that is capable of real-time image style transfer. Zhu et al [39] proposed a cycle-consistent generative adversarial network to learn the one-to-one mapping of two domain images free from the dependence on paired training data. MUNIT [14] and DRIT [21] disentangled the representation of images into a domain-invariant content code and a domain-specific style code and can generate diverse outputs from a given source domain image.

Figure 2: TextPainter contains five modules, glyph encoder to encode the text glyph, color style encoder to extract the image style features, CLIP text encoder to encode text to semantic tokens, fusion module to make the association between the text semantics and vision features, a generator to generate the text image.

The series of work on style-based generators [18, 19, 22] continues to break through the quality and stylistic diversity of generated images. Furthermore, this paper presents a novel text-image generation network that also leverages a style-based generator to achieve contextual harmony between images and text semantics.

2.2 Font Generation and Text Image Generation

Font generation aims at transferring the typographic style of one font to another. Based on Pix2Pix, some early studies [7, 24, 36] used paired data to train font generation network. Subsequent works [6, 33] were able to perform unsupervised font generation. Another research direction is few-shot font generation (FFG) [1, 15, 23, 27, 33], where the transfer of an entire font can be accomplished with a few samples of the target font.

Unlike the font generation task, style-guided text image generation takes into account not only typographic stylization but also textual stylization (e.g., color and effect), which is more challenging. SRNet [32] is the first attempt to edit the text in natural images on the word level with an end-to-end trainable style retention network. By predicting geometrical attributes of style images and the TPS (Thin-Plate-Spline) module, SWAPText[35] is able to handle severe font geometric distortions. These methods require target-style images as supervision for model training and are therefore constrained by synthetic images. To tackle this problem, TextStyle-Brush [20] proposes a self-supervised one-shot text style transfer approach that can disentangle the style of a text image of all aspects of its appearance and shows impressive results of scene text content replacement. In contrast to TextStyleBrush, which is English character-based, TextPainter is used to handle text image generation of Chinese characters which has a more complex structure in general and is therefore more challenging. Besides, APRNet [25] introduces a content-style cross-attention module and pixel sampling approach to achieve photo-realistic text image generation. Equipped with the Cross-Attention mechanism, TextPainter introduces the text comprehension module to build a bridge between text images and text semantics.

2.3 Context-aware Text Image Generation

Style-guided text image generation requires a target style image as a reference and imitates its appearance while keeping text content unaltered. Without using a target style image, some works try to model text image generation based on the context in which the text image is located. To aid the selection of fonts, colors, and sizes for designers in the process of designing web pages, Miyazono et al. [37] propose to model the font in the context of web pages using multi-task deep neural network. This approach relies on structured HTML tags and models color as a discrete classification problem, resulting in insufficient color change capability. Similarly, in order to assist the text design for posters, we propose the generation of font images based on the poster context, but with freer pixel-level output space.

3 METHOD

This section provides a brief overview of the TextPainter model's architecture, which is based on the StyleGAN framework. We present a methodology for text image generation that incorporates both local and global color harmony specifically designed for the poster design task. Additionally, we discuss how the text comprehension capability of existing multimodal models can enhance the visual design of text images. Lastly, we propose a specific padding method for image preprocessing to handle different image sizes within a single training batch.

3.1 Overview

We illustrate in Fig 2 that TextPainter can automatically generate the text image O_t on the poster by utilizing the poster's background image I_{bg}, text content c, and position b. When designing posters, color coordination between the local area of the text and the overall poster must be carefully considered, especially for text visual design. The process of generating a text image involves the color style encoder extracting information about both the local background

of the text area I_{lb} and the overall poster background I_{bg}, which is then integrated into the generator.

Moreover, we employ CLIP [34], a pre-training vision-language model, to extract text semantic information and enhance TextPainter's understanding of text semantics. This helps us create a visual design that highlights the key points of the text.

During the training phase, we adopt image-level supervision with the assumption that the presence of a single text in each image. A dataset was constructed by amassing a substantial collection of e-commerce-style poster images, and the text content and bounding boxes were labeled. In real-world applications, the text on the poster has different font sizes and lengths. In order to enable the model to perceive this information, we employ a padding method that uses background values during training.

3.2 Text-Image Generation with Local and Global Color Harmony

We use StyleGAN as the basis for our text image generation architecture, with some enhancements. To generate a text image on a poster background, we first need to render the text content into an image. Then, we use the glyph encoder to obtain the glyph feature map, which is used to generate the re-colored text image with the generator. However, this approach presents three challenges: 1) the downsampling of the initial text content image may cause the loss of structural information, 2) restoring the background of the text area based solely on the text content is an ill-posed problem, and 3) a method must be devised to ensure that the color scheme of the generated text image is harmonious with that of the poster.

To retain the glyph's structure, skip connections based on the U-net architecture are incorporated between the glyph encoder and the generator. Moreover, to enable the generator to focus only on generating the foreground text, the textual region's background image is provided as input. To address the final challenge of ensuring that the generated text images are harmonious with the poster background, several auxiliary loss functions are adopted to satisfy the constraints.

3.2.1 Global image style awareness.
TextPainter employs a style encoder to encode the background image I_{bg} into a background style vector s to perceive its style. Owing to the hierarchical structure of the generator, distinct layers control styles at various levels. The style vector s is mapped to multiple hierarchical style vectors w_i, and the AdaIN modules control w_i to operate the style generator.

3.2.2 Local patch awareness.
We previously mentioned the color style vector s, which fails to consider the position of the text. As a result, there is a possibility that text images generated at different positions may have identical colors, which is not desirable for poster visual design. To overcome this issue, we propose a solution that concatenates the poster background I_{bg} and the position mask image along the channel dimension. This concatenated image is then inputted into the style encoder. By doing so, the style encoder can obtain style information that is conditioned on the text position, thereby ensuring that the generated text images have harmonious colors with the poster background.

Figure 3: Visualization of Text comprehension module. (a) Sentence-level fusion (b) Word-level fusion.

3.3 Visual Design through Text Comprehension

Text images consist of two modalities, visual and linguistic, which means that designing visually appealing text on posters requires an understanding of the text's content. To overcome this challenge, TextPainter leverages pre-trained vision-language models such as Chinese-CLIP [34], which is the Chinese version of CLIP, to extract semantic features from the texts. These features are then merged with visual features. The extracted semantic features include both sentence-level and word/character-level information. To process these features, we propose two novel fusion methods.

3.3.1 Sentence-level Fusion.
The semantics of a sentence partly reflect the color style of the text image, making it suitable for integration into the generator via AdaIN. The complete semantic feature of the sentence is encoded by the first token z_0, after the text content c, using Chinese-CLIP. This token z_0 is then mapped and added to the style vector s to merge the features of both visual and linguistic modalities.

3.3.2 Word-level Fusion.
Highlighting the key points is the most crucial principle in poster visual design. Our approach involves utilizing word/character-level semantics to emphasize text keywords, akin to the attention mechanism. Thus, we propose the Semantic-Aware Cross-Attention plugged into the generator to align word/character-level semantic features with visual features. To the best of our knowledge, this is the first instance of utilizing text semantics to aid in generating text images.

Specifically, as shown in Figure 3, the text semantic tokens output from CLIP is denoted by $Z_t \in \mathbb{R}^{N_t \times C_1}$. Similarly, the visual tokens $Z_v^{(i)} \in \mathbb{R}^{N_i \times C_2}$ reshaped from feature map in the i-th generator layer, denoted as $X^{(i)}$. Next, take visual tokens as query and text semantic tokens as key and value, as shown in Eq (1).

$$Q = Z_v^{(i)} W_v, \quad K = Z_t W_t, \quad V = Z_t W_t \qquad (1)$$

Where both $W_v \in \mathbb{R}^{C_2 \times d_k}$ and $W_t \in \mathbb{R}^{C_1 \times d_k}$ represent linear layers that project query, key, and value to the same dimension.

Then, use the attention[29] calculation that takes both the text semantic and visual tokens, as shown in Eq (2).

$$Z = Softmax(\frac{QK^T}{\sqrt{d_k}})V + Q \qquad (2)$$

After that, $Z \in \mathbb{R}^{N_i \times d_k}$ is projected back to the dimension of the previous visual tokens $Z_v^{(i)}$, as shown in Eq (3)

$$Z_{att}^{(i)} = ZW_{out} \qquad (3)$$

Where $W_{out} \in \mathbb{R}^{d_k \times C_2}$. Finally, the result $Z_{att}^{(i)} \in \mathbb{R}^{N_i \times C_2}$ is re-shaped back to $X_{att}^{(i)} \in \mathbb{R}^{H_i \times W_i \times C_2}$, as the input of next generator layer.

Consequently, the attention calculation between character/word-level text semantics and visual features heightens the visibility of keywords in text images.

3.4 Training Strategies

3.4.1 Contextual padding. Generating text images during the training stage presents a practical challenge as texts have varying lengths, and the sizes of text fonts differ. This renders the implementation of mini-batch training infeasible. Therefore, it is essential to find a solution that can handle text images of varying sizes without distorting the font glyph.

Padding appears to be a better alternative to resizing text images to the same size during training, which is infeasible due to varying text lengths and image sizes. However, filling all images with the same pixel value leads to blurred text image edges, as discovered through extensive experiments. Fortunately, this issue can be resolved by using the background values surrounding the text images for padding, which helps overcome size constraints during training.

3.4.2 Loss function. The traditional pixel-wise supervised approach is impractical due to the challenge of annotating text masks. Thus, we adopt a weakly supervised method that utilizes text content and bounding boxes of texts in our training process. Furthermore, we employ adversarial learning to enhance the generator training.

The following loss functions are applied to efficiently train our model to generate text images with harmonious colors.

First, the reconstruction loss $\mathcal{L}_{rec}$ is defined as the L1 norm. This ensures finer details in the generated images.

$$\mathcal{L}_{rec} = \mathbb{E}\left[\frac{1}{N_t}\|I_t - O_t\|_1\right] \qquad (4)$$

Where $N_t = H_t \times W_t$, which is used as normalization due to the different sizes of text images in our dataset.

Additionally, in an effort to ensure that the generated image and ground truth possess the same style, the perceptual loss $\mathcal{L}_{per}$ is also utilized during training.

$$\mathcal{L}_{pre} = \mathbb{E}\left[\sum_i \frac{1}{M_i}\|\phi_i(I_t) - \phi_i(O_t)\|_1\right] \qquad (5)$$

Here, ϕ_i is denoted as the feature map of i-th layer in VGG[26] and $M_i = H_i \times H_i \times C_i$.

Moreover, the adversarial loss is introduced as shown in Eq (6), where D denotes the discriminator that discriminates between the generated image O_t and the real image I_t.

$$\mathcal{L}_{adv} = \mathbb{E}\left[D(O_t)\right] \qquad (6)$$

Finally, our model can be jointly optimized based on Eq (7).

$$\mathcal{L} = \lambda_1 \mathcal{L}_{rec} + \lambda_2 \mathcal{L}_{pre} + \lambda_3 \mathcal{L}_{adv} \qquad (7)$$

3.4.3 Loss weighting. GAN training is often plagued by instability, with the discriminator dominating the learning process at the start. To address this issue, greater weight is assigned to the reconstruction loss during early training, and the dynamic adjustment is made based on the following strategy as the generator enhances.

$$\lambda_1 = r^n \qquad (8)$$
$$\lambda_3 = 1 - \lambda_1 \qquad (9)$$

Where n denotes the number of the current training epoch and r represents the rate of L_{rec}, set to 0.85 in our experiments. This method aims to stabilize training and prevent the discriminator from dominating the learning process.

Figure 5: A partial examples of the PosterT80K dataset.

4 POSTER-T80K DATASET

Posters play a crucial role in effectively conveying information and often exhibit visually-rich styles. To aid in training and testing the TextPainter network, we collected poster images from Chinese shopping sites, each with a resolution of 513x750. Images that lacked copywriting were filtered out. We collected 165,494 images and filtered out the images without text leaving 117,624 images. The training set and test set are split into 106,009 and 11,615 respectively. The collected posters are annotated with sentence-level bounding boxes and the text string content of each sentence. In order to remove the text from the image and preserve the background, given the labeled text box, we use the text erasing method [17] to perform the erasure of the text on the image.

Figure 5 shows a sample set of poster images in the dataset. Since the poster images and text are manually designed, unlike synthetic text image data usually used for font generation, Poster-T80K better reflects the challenges of real-world text image generation (e.g. irregular gradients and keyword highlight). And Chinese characters have a complex glyph structure compared to English, compared with Book Cover Dataset [4].

Figure 4: Generated text images (paste back to the background images) using TextPainter. (Top) The different results generated by TextPainter given different contextual background images. (Bottom) Adaptive change results generated by TextPainter based on different text contents.

5 EXPERIMENTS

5.1 Implementation Details

Based on StyleGAN, TextPainter consists of the generator, the glyph encoder, the style encoder, and fusion modules. The glyph and style encoders are implemented using ResNet-34 [11]. The word/character-level fusion module is only plugged into the last two layers of the generator. The text encoder is loaded from the base version of Chinese-CLIP and it performs character-level tokenization for the sentence. After encoding the text, only the first 16 tokens of the token sequence are utilized as input, which is due to the maximum length of the texts in our dataset.

5.2 Evaluation Metrics

Three commonly-used metrics are adopted for image generation to evaluate the performance of our model: Frechet Inception Distance (FID)[12], structural similarity index measure (SSIM)[31], Peak signal-to-noise ratio (PSNR). The ground-truth text images and the generated results are compared to calculate these metrics.

5.3 Comparsion results

Baseline methods. To the best of our knowledge, no prior work has directly addressed our task. Therefore, a modified version of WebFont [37] is our initial baseline approach. Moreover, we also implement two traditional methods as our baselines. The details of these baselines will be elaborated below.

- **Base on the classification.** WebFont [37] is a classification-based approach that uses information such as images to predict text attributes, which is implemented by ourselves.
- **Base on the color contrast.** The approach extracts the main colors respectively for both local and global images. Then, the color with the highest contrast between the global and local ones is selected from the global main colors.
- **Base on the retrieval.** This method utilizes color histograms extracted from both global and local background images of the poster as features, with text color as the label.

Baseline Result. Table 1 gives the quantitative comparison results of the different methods and on the test set of the dataset that we proposed. The experimental results indicate that the color contrast-based method produces the lowest quantitative results on the test data, possibly because this method relies on human observation-based rules, instead of learning patterns from data. The classification-based methods also show poor performance, as such methods require quantification of the RGB values of colors, which often results in uneven distribution of colors in the color space, making the classification results easily biased towards specific categories. Both retrieval-based methods and TextPainter have obtained favorable quantitative outcomes. Nonetheless, retrieval-based methods necessitate the annotation of text colors for training, whereas TextPainter solely requires bounding boxes. Besides, TextPainter is a GAN-based method with a disadvantage in the FID compared to other methods that are based on graphic rendering. This issue is proven in our result, therefore TextPainter achieves better results

Table 1: The quantitative results of different baseline methods and TextPainter.

Method	FID↓	SSIM↑	PSNR↑
Classification [37]	23.32	0.6801	32.46
Color Contrast	25.43	0.6939	32.78
Retrieval	**17.38**	0.6928	32.78
Ours	<u>18.53</u>	**0.7042**	**33.49**

in terms of SSMI and PSNR. Moreover, as depicted in Figure 4, our method is based on text comprehension, which is capable of highlighting the keywords in the text. Overall, our method's primary advantage is its capability to infer visually emphasized content from the text semantic.

5.4 Analysis

Visual context-aware capability. We investigate the capability of TextPainter to comprehend visual contexts by utilizing different poster backgrounds as the style encoder of the input while maintaining the text input as "only one for one person." In Figure 4 Top, the results illustrate that TextPainter can detect contextual changes and produce text visuals with harmonized colors, leveraging the capabilities of our style encoder.

Text semantics-aware capability. In addition, we explore whether TextPainter can comprehend text semantics by fixing the poster background and text position, while varying the text content. As shown in Figure 4 (Bottom), the outcomes reveal that TextPainter can emphasize distinct keywords in the generated text visuals for different text contents. It is noteworthy that TextPainter is capable of understanding word order (1st and 2nd columns, 3rd and 4th columns) and synonym substitutions (the last two columns).

Effectiveness of style encoder. The style encoder is utilized to extract the color style of the poster background. However, as shown in the generation block in Figure 1, in order to generate text in the background of the text area, the local background I_lb and feature map are concatenated. This raises concern that the style encoder may fail to function since the generator may directly learn the color style from the local background I_lb and disregard the input from the style encoder. To address this issue, we conducted experiments to validate our concern, as illustrated in Figure 6. The qualitative results indicated that the color style of the text image is not affected by the local background I_lb, but rather by the input from the style encoder. This is due to the fact that TextPaint only blends the text and background in the final block, where the color style of the text image is already established in the previous blocks.

Color style interpolation. In order to investigate additional properties of TextPainter, we performed interpolation experiments on styles extracted by the style encoder. We utilized two different styles extracted from two distinct poster backgrounds for linear interpolation, which served as input to the generator. Figure 7 demonstrates that the color of the text image can transition smoothly between the two styles. The findings of this study demonstrate that the TextPainter color style exhibits significant continuity in the feature space. Consequently, there is an opportunity to further explore color editing techniques in future research.

Figure 6: The qualitative result of the color style encoder: "Global" means the poster background input in the color style encoder. "Local" represents the local background used by the input generator for background blending.

Figure 7: The result of the interpolation experiment where the color style vector is linearly interpolated between the source and target.

The impact of the artifact after text erasing. The proposed dataset comprises pairs of a poster image ground truth and a background image. In particular, the poster background image is generated by the inpainting model [17] to erase the text image in the poster, which may result in artifacts that are not visible to the naked eye and information leakage. To assess the impact of artifacts, a small subset of clean poster backgrounds without artifacts have been collected. For the purpose of comparison, we also created the same background images with artifacts. Finally, compare the results of generating using the two different background images. As shown in Table 1, the qualitative results in Figure 8 indicate that the effect of artifacts is extremely slight.

Table 2: The quantitative results of test data with artifact and test data without artifact.

Method	FID↓	SSIM↑	PSNR↑
w/o artifact	21.49	0.6943	32.54
w/ artifact	21.23	0.6948	32.81

Figure 8: Qualitative ablation experiment results of TextPainter's different modules. (a) Ours. (b) w/o Sentence-level Fusion. (c) w/o Word-level Fusion. (d) w/o Text Encoder. (e) w/o Mask. (f) w/o Style Encoder.

Table 3: The quantitative results of ablation studies.

Method	FID↓	SSIM↑	PSNR↑
w/o Style Encoder	25.76	0.6747	32.20
w/o Mask	19.45	0.7011	33.01
w/o Text Encoder	20.52	0.6839	32.42
w/o Sentence-level Fusion	18.98	0.7056	33.28
w/o Word-level Fusion	20.21	0.6991	32.87
Ours	**18.53**	**0.7042**	**33.49**

5.5 Ablation Studies

Table 3 and Figure 8 provide an ablation study evaluating the effects of the color style encoder, text encoder, position mask, and the fusion of text semantics at different granularities, respectively. The term "w/o Style Encoder" refers to the absence of a color style encoder. Similarly, "w/o Mask" denotes the omission of the position mask of text, whereas "w/o Text Encoder" signifies the non-existence of a text comprehension module. Additionally, "w/o Sentence-level Fusion" denotes the lack of cross-attention, and "w/o Word-level Fusion" implies the absence of sentence-level tokens.

Color style encoder. Based on the result, the ablation of the color style encoder resulted in a noteworthy decline in performance. The main reason is that the absence of a style color encoder model makes it impossible to extract the color style from the visual context of the poster and guide the generation of globally and locally harmonious color text images.

Position mask. Obviously, as shown in Figure 8, the lack of masks makes the style encoder only focus on the global background, which can lead to poor color harmony between the text image and the local background.

Text Encoder. The removal of the text understanding module, either partially or completely, significantly weakens the emphasis on keywords in the text, as illustrated in (b)(c)(d) of Figure 8. This finding serves as evidence that the inclusion of the text understanding module can markedly enhance our method.

5.6 User study

The evaluation of poster design emphasizes aesthetics. Due to the difficulty of quantifying aesthetics, we adopted a user study to evaluate different methods. Specifically, we randomly sampled 32 groups of poster images generated from the test results of different method. Then we presented participants with randomly-ordered generated images from distinct methods. 30 users were asked to choose the poster images of the most aesthetic appeal in the group. The aesthetic result of the user study is shown in Figure 9, where numbers represent the percent of times users chose. According to the results, the highest number of users voted for TextPainter, which indicates that our TextPainter, based on the text understanding module, can automatically generate more attractive posters.

6 CONCLUSION

In this paper, we present the task of text image generation on the poster for the first time. Our method aims to automatically generate clear, harmonious, colorful, and creative text images on posters based on image contextual and text content, like a designer. Experiments validate the ability of our method to model contextual visual and textual semantics and to generate visually appealing and meaningful text images. The extent experiment demonstrates the potential of the method in scalable training through a self-supervised training approach. Finally, we collect a novel Chinese poster dataset with annotation and hope our work will encourage future research on multi-modal text image generation.

Figure 9: The User study results on the aesthetics of different methods.

ACKNOWLEDGMENTS

REFERENCES

[1] Samaneh Azadi, Matthew Fisher, Vladimir G. Kim, Zhaowen Wang, Eli Shechtman, and Trevor Darrell. 2017. Multi-Content GAN for Few-Shot Font Style Transfer. *computer vision and pattern recognition* (2017).

[2] Dan S Bloomberg. 2008. Color quantization using modified median cut. *Leptonica http://www. leptonica. com/paperi/mediancut. pdf* (2008).

[3] Yunning Cao, Ye Ma, Min Zhou, Chuanbin Liu, Hongtao Xie, Tiezheng Ge, and Yuning Jiang. 2022. Geometry Aligned Variational Transformer for Image-conditioned Layout Generation. In *ACM Multimedia*. ACM, 1561–1571.

[4] Roxzanne Van Eyk. 2008. Judging a book by its cover. *Journal of Marketing* (2008).

[5] Yiqi Gao, Xinglin Hou, Yuanmeng Zhang, Tiezheng Ge, Yuning Jiang, and Peng Wang. 2022. CapOnImage: Context-driven Dense-Captioning on Image. *CoRR* abs/2204.12974 (2022). https://doi.org/10.48550/arXiv.2204.12974 arXiv:2204.12974

[6] Yiming Gao and Jiangqin Wu. 2018. CalliGAN: Unpaired Mutli-chirography Chinese Calligraphy Image Translation. *asian conference on computer vision* (2018).

[7] Yiming Gao and Jiangqin Wu. 2020. GAN-Based Unpaired Chinese Character Image Translation via Skeleton Transformation and Stroke Rendering. *national conference on artificial intelligence* (2020).

[8] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. 2015. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576* (2015).

[9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. *neural information processing systems* (2014).

[10] Shunan Guo, Zhuochen Jin, Fuling Sun, Jingwen Li, Zhaorui Li, Yang Shi, and Nan Cao. 2021. Vinci: An Intelligent Graphic Design System for Generating Advertising Posters. *human factors in computing systems* (2021).

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. *arXiv: Computer Vision and Pattern Recognition* (2015).

[12] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2017. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 6626–6637. https://proceedings.neurips.cc/paper/2017/hash/8a1d694707eb0fefe65871369074926d-Abstract.html

[13] Xun Huang and Serge Belongie. 2017. Arbitrary Style Transfer in Real-time with Adaptive Instance Normalization. *Learning* (2017).

[14] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. 2018. Multimodal Unsupervised Image-to-Image Translation. *european conference on computer vision* (2018).

[15] Yaoxiong Huang, Mengchao He, Lianwen Jin, and Yongpan Wang. 2020. RD-GAN: Few/Zero-Shot Chinese Character Style Transfer via Radical Decomposition and Rendering. *european conference on computer vision* (2020).

[16] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. 2016. Image-to-Image Translation with Conditional Adversarial Networks. *computer vision and pattern recognition* (2016).

[17] Gangwei Jiang, Shiyao Wang, Tiezheng Ge, Yuning Jiang, Ying Wei, and Defu Lian. 2022. Self-Supervised Text Erasing with Controllable Image Synthesis. *arXiv preprint arXiv:2204.12743* (2022).

[18] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. 2021. Alias-Free Generative Adversarial Networks. *arXiv: Computer Vision and Pattern Recognition* (2021).

[19] Tero Karras, Samuli Laine, and Timo Aila. 2018. A Style-Based Generator Architecture for Generative Adversarial Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2018).

[20] Praveen Krishnan, Rama Kovvuri, Guan Pang, Boris Vassilev, and Tal Hassner. 2021. Textstylebrush: transfer of text aesthetics from a single example. *arXiv preprint arXiv:2106.08385* (2021).

[21] Hsin-Ying Lee, Hung-Yu Tseng, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang. 2018. Diverse Image-to-Image Translation via Disentangled Representations. *International Journal of Computer Vision* (2018).

[22] Jaakko Lehtinen, Janne Hellsten, Miika Aittala, Tero Karras, Samuli Laine, Timo Aila, Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. 2019. Analyzing and Improving the Image Quality of StyleGAN. *computer vision and pattern recognition* (2019).

[23] Wei Liu, Fangyue Liu, Fei Din, Qian He, and Zili Yi. 2022. XMP-Font: Self-Supervised Cross-Modality Pre-training for Few-Shot Font Generation.

[24] Pengyuan Lyu, Xiang Bai, Cong Yao, Zhen Zhu, Tengteng Huang, and Wenyu Liu. 2017. Auto-Encoder Guided GAN for Chinese Calligraphy Synthesis. *international conference on document analysis and recognition* (2017).

[25] Yangming Shi, Haisong Ding, Kai Chen, and Qiang Huo. 2022. APRNet: Attention-based Pixel-wise Rendering Network for Photo-Realistic Text Image Generation. *arXiv preprint arXiv:2203.07705* (2022).

[26] Karen Simonyan and Andrew Zisserman. 2014. Very Deep Convolutional Networks for Large-Scale Image Recognition. *computer vision and pattern recognition* (2014).

[27] Licheng Tang, Yiyang Cai, Jiaming Liu, Zhibin Hong, Mingming Gong, Minhu Fan, Junyu Han, Jingtuo Liu, Errui Ding, and Jingdong Wang. 2022. Few-Shot Font Generation by Learning Fine-Grained Local Styles.

[28] Praneetha Vaddamanu, Vinay Aggarwal, Bhanu Prakash, Reddy Guda, Balaji Vasan Srinivasan, and Niyati Chhaya. 2022. Harmonized Banner Creation from Multimodal Design Assets.

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. *neural information processing systems* (2017).

[30] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. 2017. High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs. *computer vision and pattern recognition* (2017).

[31] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Process.* 13, 4 (2004), 600–612. https://doi.org/10.1109/TIP.2003.819861

[32] Liang Wu, Chengquan Zhang, Jiaming Liu, Junyu Han, Jingtuo Liu, Errui Ding, and Xiang Bai. 2019. Editing Text in the Wild. *acm multimedia* (2019).

[33] Yangchen Xie, Xinyuan Chen, Li Sun, and Yue Lu. 2021. DG-Font: Deformable Generative Networks for Unsupervised Font Generation. *computer vision and pattern recognition* (2021).

[34] An Yang, Junshu Pan, Junyang Lin, Rui Men, Yichang Zhang, Jingren Zhou, and Chang Zhou. 2022. Chinese CLIP: Contrastive Vision-Language Pretraining in Chinese. *arXiv preprint arXiv:2211.01335* (2022).

[35] Qiangpeng Yang, Jun Huang, and Wei Lin. 2020. SwapText: Image Based Texts Transfer in Scenes. *computer vision and pattern recognition* (2020).

[36] Jiang Yue, Zhouhui Lian, Yingmin Tang, and Jianguo Xiao. 2019. SCFont: Structure-Guided Chinese Font Generation via Deep Stacked Networks. *national conference on artificial intelligence* (2019).

[37] Nanxuan Zhao, Ying Cao, and Rynson WH Lau. 2018. Modeling fonts in context: Font prediction on web designs. In *Computer Graphics Forum*, Vol. 37. Wiley Online Library, 385–395.

[38] Min Zhou, Chenchen Xu, Ye Ma, Tiezheng Ge, Yuning Jiang, and Weiwei Xu. 2022. Composition-aware Graphic Layout GAN for Visual-Textual Presentation Designs. In *IJCAI*. ijcai.org, 4995–5001.

[39] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*. 2223–2232.

A POSTERT80K DATASET

This section provides a comprehensive introduction to the PosterT80K dataset. To begin with, we present a statistical analysis of the dataset. Next, we describe the data processing methods that we employed. The raw dataset comprises 117,624 Chinese poster images that we collected from e-commerce websites. Each poster image has a size of 513x750 and contains multiple texts, which total 376,844 text images. We carefully labeled the content and position of each text image by manual annotation. Specifically, we denoted the content as c and the position as $p = (x, y, w, h)$. After dataset processing, the training set contains 106,009 posters and 148,891 text images, while the testing set comprises 11,615 posters and 16,603 text images.

A.1 Data distribution analysis

Text content analysis. The distribution of text-related features was the primary focus of our analysis. Figure 10 illustrates that nearly 90% of the text lines in the dataset contain less than 12 characters, while almost 99% contain less than 22 characters. These findings suggest that most of the text lines in the posters are short sentences. Besides, the top 20 frequently appearing characters are identified and recorded in Table 4. Since e-commerce posters have a specific purpose of highlighting selling points and prices, numbers and some Chinese characters, such as '抢' (rush), '立' (stand), and '购' (buy), frequently feature in the text. Figure 11 reveals that approximately 80% of posters contain 1 to 5 texts, and almost 99% of posters contain 1 to 12 texts.

Figure 10: The distribution of text length in the raw dataset. (horizontal axis) text length. (vertical axis) probability density.

Figure 11: The distribution of the number of text lines that posters contain in the raw dataset.(horizontal axis) the number of text lines. (the vertical axis) the probability density.

Figure 12: The distribution of the height of the text images in the raw dataset. (horizontal axis) the height. (vertical axis) the probability density.

	Character	Count		Character	Count
1	'0'	89048	11	'3'	25268
2	'1'	78131	12	'即' (immediately)	23265
3	' '	65471	13	'元' (yuan)	20906
4	'2'	41279	14	'一'	20123
5	'9'	29215	15	'买' (buy)	19792
6	'抢' (rush)	28132	16	'价' (price)	18270
7	'立' (stand)	28035	17	'4'	14641
8	'>'	27069	18	'限' (limit)	14599
9	'购' (buy)	25625	19	'送' (give)	14555
10	'5'	25622	20	'新' (new)	14265

Table 4: The top 20 characters that appear in the text of the dataset.

Distribution of text image size. Although the size of the image of the poster is fixed, the size of the text image varies, which complicates the training of the model. To address this issue, we performed an analysis of the text image sizes. The distributions of height and width of the text image are presented in Figure 12 and Figure 13, respectively. The results show that approximately 94% of the height values are within the range of 16 to 99, exhibiting a high level of concentration, while the width values display a broader distribution ranging from 25 to 513. Furthermore, as depicted in Figure 14, the aspect ratio is also highly concentrated, with approximately 95% of the values falling between 1.0 and 11.0, which is consistent with the distribution of our text length.

Figure 13: The distribution of the width of the text images in the raw dataset, with the horizontal axis representing the width and the vertical axis representing the probability density.

Figure 14: The distribution of the aspect ratio of the text images in the raw dataset. (horizontal axis) the aspect ratio. (vertical axis) the probability density.

Figure 15: The architecture of the baseline method based on classification.

Layer	Configurations	Output
Input	Glyph Feature Map	$512 \times \frac{H}{32} \times \frac{W}{32}$
Conv	Conv2D	$512 \times \frac{H}{32} \times \frac{W}{32}$
Block1	StyleConv	$512 \times \frac{H}{32} \times \frac{W}{32}$
	StyleConv	
	LeakyReLU	
	RGBConv	
Block2	Cross-Attention	$256 \times \frac{H}{16} \times \frac{W}{16}$
	Upsample	
	StyleConv	
	StyleConv	
	LeakyReLU	
	RGBConv	
Block3	Cross-Attention	$128 \times \frac{H}{8} \times \frac{W}{8}$
	Upsample	
	StyleConv	
	StyleConv	
	LeakyReLU	
	RGBConv	
Block4	Cross-Attention	$64 \times \frac{H}{4} \times \frac{W}{4}$
	Upsample	
	StyleConv	
	StyleConv	
	LeakyReLU	
	RGBConv	
Block5	Cross-Attention	$32 \times \frac{H}{2} \times \frac{W}{2}$
	Upsample	
	StyleConv	
	StyleConv	
	LeakyReLU	
	RGBConv	
Block6	Upsample	$3 \times H \times W$
	StyleConv	
	StyleConv	
	LeakyReLU	
	RGBConv	

Table 5: The architecture of the generator.

A.2   Data preprocessing

According to the data distribution analysis, the dataset was first filtered to exclude outlier values in order to facilitate model training, after which the dataset was divided. The filtering process involved the following steps:

- Removing posters containing more than 5 text images.
- Removing text images with aspect ratios greater than 11.
- Removing text images with heights outside of the range of 30 to 100 or widths outside of the range of 50 to 450.
- Removing text images outside of the content length range of 1 to 11 characters.

Finally, the dataset was then divided based on poster images.

B   MODEL ARCHITECTURE

The style encoder and glyph encoder of TextPainter are both implemented by ResNet34, with the dimension of the style feature vector being 512. Besides, the model architectures of the generator are shown in Table 5. Furthermore, AdaIN is implemented in StyleConv by a Linear layer and a Conv2DMod layer, while RGBConv is

implemented by a StyleConv, an UpSample layer, and a Blur layer, consistent with the implementation of StyleGAN. Finally, we use the base version of Chinese-CLIP text encoder.

C   THE IMPLEMENTATION OF BASELINES

C.1   Base on the classification.

WebFont [37] is a classification-based approach that uses information such as images to predict text attributes. As our task only relies on the poster background and text, we implemented WebFont as shown in Figure 15. Specifically, we quantized the RGB color values into 26 categories based on the WebFont settings and then used the global and local style encoders to extract features, which were fused to predict the text color.

C.2   Base on the color contrast.

Firstly, extract five theme colors from the overall background of the poster. Secondly, extract one theme color from the local background of the text position. Finally, use the color with the highest contrast with the local theme color among the selected five global theme colors as the text color. The Modified Median Cut Quantization algorithm [2] is used to extract the theme colors.

C.3   Base on the retrieval.

Firstly, RGB color histograms are separately extracted from the global and local backgrounds and concatenated as the text image feature. The histogram dimension for each single color component is 128, and the total feature dimension is 768. Then, the RGB color of the text serves as the label and is used to establish a retrieval database together with the text image feature. During the inference stage, the same feature extraction method is used, and the most similar sample is retrieved from the database to predict text color.

Figure 16: The qualitative results of the impact of the two different padding methods.

D   THE CONTEXTUAL PADDDING

The use of padding is necessary to conform text images of varying sizes to a batch and to avoid any negative impact on font size resulting from resizing. However, as shown in Figure 16, our experimentation has revealed that fixed padding values cause distortion in the generated local background. To overcome this issue, we propose an alternative solution that utilizes the background values surrounding the text area for padding. As demonstrated in Figure 16, this approach results in a significant improvement in the overall effect, enabling a better blend between the local and surrounding backgrounds.