DS-Fusion: Artistic Typography via Discriminated and Stylized Diffusion

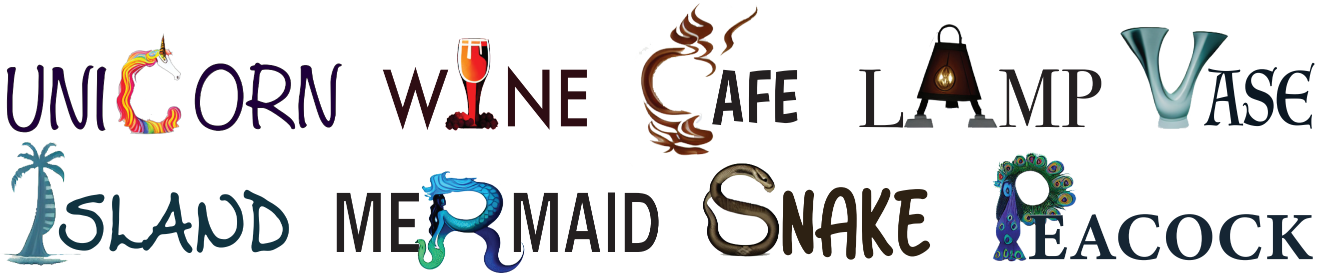

Figure 1. Single-letter artistic typographies generated *fully automatically* by our network. Each result is produced from a prompt consisting of a word, e.g., "unicorn", and a letter in the word, e.g., "C", to be stylized based on the semantics of the word and the input font.

Abstract

We introduce a novel method to automatically *generate an* artistic typography *by* stylizing *one or more letter fonts to visually convey the semantics of an input word, while ensuring that the output remains readable. To address an assortment of challenges with our task at hand including conflicting goals (artistic stylization vs. legibility), lack of ground truth, and immense search space, our approach utilizes large language models to bridge texts and visual images for stylization and build an* unsupervised *generative model with a diffusion model backbone. Specifically, we employ the denoising generator in Latent Diffusion Model (LDM), with the key addition of a CNN-based* discriminator *to adapt the input style onto the input text. The discriminator uses rasterized images of a given letter/word font as real samples and output of the denoising generator as fake samples. Our model is coined DS-Fusion for discriminated and stylized diffusion. We showcase the quality and versatility of our method through numerous examples, qualitative and quantitative evaluation, as well as ablation studies. User studies comparing to strong baselines including CLIP-Draw and DALL-E 2, as well as artist-crafted typographies, demonstrate strong performance of DS-Fusion.*

1. Introduction

We explore the artistic-creative potential of automated generative processes modeled by modern neural networks. Specifically, we are interested in the generation of *artistic typography*. According to Wikipedia, typography is the art and technique of arranging type to make written language legible, readable, and appealing when displayed. Artistic typography represents a style of typography that goes beyond the basic function of conveying information through text and seeks to create a visual impact on the reader. It involves using typography as a form of artistic expression and allows designers to create eye-catching typographic designs that express a message visually and creatively.

In this paper, we aim to automatically generate an artistic typography by *stylizing* one or more letter fonts, to visually convey the semantics of an input word, while ensuring that the output typography is readable; see Figure 1 for such examples referred to as "word-as-image" [3, 27, 34, 10]. It is an arduous task to combine semantics and text in a legible and artistic manner for several reasons. First, the goal of incorporating the aesthetics of a style in an abstract and creative way into a letter or word can conflict with the desire to maintain readability of the original word/letter. Second, what a good artistic typography is can be a subjective matter. Without a universally accepted "ground truth", a viable learning approach will have to be *unsupervised*. Last but not least, semantics can be depicted in numerous ways. For instance, to indicate the presence of a lion, one can use the entire face, the tail, or the whole animal. There is a vast range of lion images, icons, and shapes that are accessible, making it nearly impossible to manually search, deform, and substitute them. While experienced artists and designers are capable of producing beautiful semantic typography, obtaining reasonable results for ordinary users and hobbyists without proper assistive tools are out of reach.

To address all the above challenges, we resort to recently popularized large language models [21, 19] to bridge texts and visual images for stylization and build our unsupervised

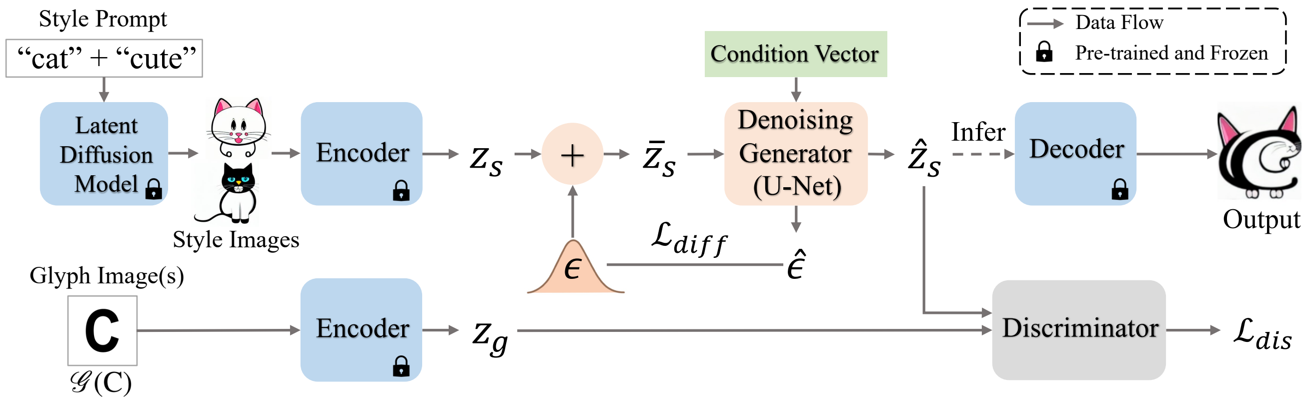

Figure 2. The pipeline of DS-Fusion, which takes as input a style prompt and a glyph image. The style images are generated according to the style word and attribute. DS-Fusion first utilizes a latent diffusion process [21] to construct the latent space of the given style and then introduces a discriminator to blend the style into the glyph shape. The parameters of a module are pre-trained and frozen if there is an icon of a lock on the bottom right. The "+" module denotes the iterative noise injection process of diffusion models.

generative model for artistic typography on Latent Diffusion [19]. Specifically, we employ the *denoising generator* in Latent Diffusion Model, with the key addition of a CNN-based *discriminator* to adapt the input style onto the input text (Figure 2). The discriminator uses rasterized images of a given letter/word font as real samples and output of the denoising generator as fake samples. To obtain images for the denoising generator to guide the letter stylization, we generate 25 style images from the input word, again, with Latent Diffusion. The selection of this number aims to ensure an adequate number of instances for the diffusion model to extract underlying features and attain diversity in the outputs. We fine-tune the denoising generator on these images using the diffusion loss based on the style images and the discriminator loss.

Our model is coined *DS-Fusion* for discriminated and stylized diffusion. While the core idea is quite simple, it is among the first to integrate adversarial learning and diffusion in a single framework. By utilizing the powerful generation capabilities of diffusion models and employing a discriminator as a critic, we ensure that the produced artistic typography remains true to the input font. Figure 1 shows some results generated by DS-Fusion *fully automatically*.

We showcase the effectiveness of DS-Fusion for generating artistic typography through numerous experiments. Our approach produces visual results that demonstrate its generality and versatility in accommodating different semantics, letters, and artistic styles. We report quantitative evaluations and ablation studies to assess the contribution of individual components. Additionally, we have conducted user studies to assess the quality of our automatically generated typography results. These studies reveal that in about 42% of the cases, our results were favored over or considered equally good as results produced by professional artists, and in close to 50% of cases, our method outperforms the state-of-the-art alternatives, including DALL-E 2 [20], a strong baseline that had been trained on significantly more images than LDM. It is worth noting that we did not choose specific inputs catering to DS-Fusion for these comparisons. Instead, we performed a Google image search on "artistic typography" and extracted a suitable subset of artist-generated results to come up with inputs for both user studies.

2. Related work

Learning fonts. There are numerous techniques to study, design, and stylize fonts. Campbell and Kautz [4] utilized an algorithm to learn a font manifold from the polyline representation of glyph outlines. By exploring this manifold, new fonts can be obtained, or existing fonts can be interpolated to achieve a desired effect. Balashova et al. [2] proposed an approach that uses a stroke-based geometric model for glyphs and a fitting procedure to reparametrize arbitrary fonts to the representation, which is again estimated through a manifold learning technique that estimates a low-dimensional font space. More recently, Wang et al. [29] proposed a dual-modality learning scheme to synthesize vector fonts, which are refined with glyph images using differentiable rasterization.

Font stylization. Efforts have also been made to stylize fonts to enhance their artistic and aesthetic appeal. For instance, Azadi et al. [1] proposed a conditional-GAN [18, 11] to generate glyph images with different font and texture styles that match a given template. Berio et al. [3] proposed to segment a font's glyphs into a set of overlapping and intersecting strokes to generate artistic stylizations. To accomplish context-aware text image stylization and synthesis, Yang et al. [31, 33, 32] proposed a style transfer method with the ability to preserve legibility. Nonetheless, artistic typography surpasses mere manipulation of fonts and often entails imbuing letters with a semantic meaning.

Semantic and artistic typography. Semantic typography involves adding certain elements to a text to emphasize certain aspects, communicate a message, or highlight a property. There have been several endeavors to integrate multiple elements in the creation of a logo, text, or other design elements. One example of this is through the use of collage-based techniques to fill a letter by incorporating semantic elements within it [14, 23, 6, 35]. The concept of legible calligrams [36] focuses on solving an inverse problem by placing a word or group of letters into a semantic shape. When our approach is applied to an entire word as the input (as shown in Figure 5), it may yield results that resemble legible calligrams. However, our problem statement is entirely distinct, and our approach to solving it is significantly dissimilar because our method is learning-based and not limited to a predetermined template shape that dictates the letter placement and deformation.

The works that are most closely related to ours are those that attempt to alter a letter or a set of letters to achieve a specific semantic meaning, whether through replacement, deformation, or texturization. For instance, Zhang et al. [34] propose a semi-manual technique that involves manually dividing a letter into sections, fitting semantically related shapes to those sections, and performing post-processing to eliminate artifacts. However, the effectiveness of this method is heavily reliant on the accuracy of manual letter segmentation and the use of predefined shapes, which can impact the quality of the results. Trick or treat [27] is an attempt to replace a letter with an icon by identifying an icon that closely matches the letter from a joint embedding of letters and icons. The chosen icon is then slightly deformed to better represent the letter. However, this method requires the existence of an icon that closely resembles the letter in order to produce satisfactory results. Instead, we argue that a more effective solution would involve learning how to *generate* the desired semantic typography by creating letters or words that convey a particular semantic or aesthetic feature in a subtle yet effective manner, as depicted in Figure 4.

Text-based generative design. Large Language Models such as BERT [7] significantly advance the understanding of human language, which makes text-based generation tasks much easier. With the emergence of powerful models that can establish connections between natural language and images, such as CLIP [19], several downstream tasks have benefited from these models, including mesh and image editing, stylization, and generation [16, 13, 28, 17]. Of particular relevance to our work, CLIPDraw [8] aims to produce SVG-format drawings by first rasterizing them using DiffVG [15], and then utilizing CLIP for evaluation. Recently, there has been a surge in popularity of diffusion models [24, 9, 25, 26], including stable diffusion [21], which has produced impressive text-to-image results. Our approach utilizes latent diffusion [21] to encode and decode glyph and style images, and employs BERT to condition the denoising process (Figure 2). To ensure both glyph and style images are respected, we utilize a discriminator to combine adversarial learning and diffusion, making it one of the first of its kind. In contrast to Diffusion-GAN [30] which uses a discriminator to distinguish a diffused real image from a diffused fake image at all steps, our discriminator is designed to preserve glyph structures in stylized images as one component of our optimization objectives.

Semantic Typography.

In concurrent work, Word-as-Image Semantic Typography (ST) [10] stylizes a letter through a semantic-aware *font deformation*; see Figure 10. To guide the deformation, ST uses a pre-trained Stable Diffusion model along with losses to preserve the font structure. Our approach differs from ST in multiple ways. We focus on extracting salient features of a style and applying them to a glyph shape and ensure legibility using a discriminator rather than deforming a font. This allows us to incorporate multiple relevant colors to semantic and stylistic attributes in raster form, while the results produced by ST remain single color in vector form. In addition, we can stylize multiple letters together as a single shape (Figure 5) while ST deforms each letter individually.

3. Method

Our method takes as input a *style prompt*, in the form of texts, and a *glyph* to be stylized, as a raster image. As we focus on generating artistic typographies in this work, the input glyph represents a graphical form of a letter or a word, e.g., in a particular font. The output of our method is a stylized version of the glyph based on the style prompt, which consists of a style *word*, and optionally a style *attribute*. The style word is a noun specifying an object or activity whose semantics are embedded into the resulting typography, while the style attribute provides a further characterization. Compelling typographies can be generated when the input glyph letters are part of the style word, e.g., “C” in “cat”, and the stylized glyph is displayed within the word, resulting in a “*word-as-image*” [3, 27, 34, 10]; see Figure 1.

We express an input by putting the style prompts in quotations and use the function $\mathscr{G}(\cdot)$ to denote a mapping from letter contents to glyph images. For example, the running example in Figure 2 has the input, “cat” + “cute” + $\mathscr{G}$(C).

3.1. Overview

To generate artistic typography, we first turn the input style word into a visual representation by employing a Latent Diffusion model [21] to obtain a set of twenty-five style images. The generative task then amounts to embedding the input semantics and the style images into a new, artistically stylized glyph, based on the input font. In the absence of any target images to provide direct supervision, we not only need a suitable feature representation for both the glyph and the style prompts, but also a means to evaluate the stylized results implicitly and effectively.

Utilizing the Latent Diffusion as our architecture backbone, we introduce the key idea of incorporating a *discriminator*, which guides the Diffusion model to produce images that fully blend styles into the input glyph images. Specifically, the diffusion first constructs the latent space of the given styles and outputs plausible latent codes, then the discriminator aims to distinguish between synthesized results and vanilla glyphs. Figure 2 shows our method pipeline, with details to follow.

3.2. The Style Latent Space

To generate a diversity of artistic typography images, we need to construct a latent space of given styles. Following the LDM, the style images are first fed into an encoder and then passed through a diffusion process. The encoder outputs the features maps z_s. A Gaussian noise $\epsilon \in \mathcal{N}(0, \mathcal{I})$ is applied on z_s to obtain $\bar{z}_s$, following an iterative noise injection process introduced by diffusion models.

A *denoising generator*, with a U-Net [22] architecture conditioned on an encoded vector by BERT, performs the denoising process for $\bar{z}_s$. It aims to predict the added noises $\hat{\epsilon}$ from $\bar{z}_s$ and then denoise $\bar{z}_s$ to $\hat{z}_s$. The prompt serves as a condition to the diffusion process, to enforce the fusion of the styles and glyphs into our desired images. Technically, any text-conditioning vector fed to the denoising generator, even a random one, could be applied since over time, the generator will fine-tune it. However, in our current implementation, we combine the input style prompt with the designated input letters as a prompt to BERT, and use the BERT-encoded vector for text conditioning, with the intent to "warm start" the generator for faster convergence. Finally, we obtain $\bar{z}_s$ for sampling styles and $\hat{z}_s$ for generating our desired images. During training and inference, $\hat{z}_s$ will be sent into the discriminator and the decoder, respectively.

The pre-trained encoder and decoder from Latent Diffusion [21] are used in our network; they are frozen during both the training and testing processes. The encoder and decoder were trained with 400M images and can extract high-quality image features. We employ a diffusion loss which measures the mean square error (MSE) between the predicted noise $\hat{\epsilon}$ and the input noise ϵ: $\mathcal{L}_{diff} = ||\hat{\epsilon} - \epsilon||_2^2$.

3.3. The Discriminator

Since there are no target images available in this task, the generated results can only be supervised implicitly. It is inappropriate to directly align the generated images with glyph images because the former is highly textured and might have a displacement of glyph outlines. To this end, we propose to employ a discriminator in the style of GANs as the examiner. Different from vanilla GANs, the discriminator here takes input as feature maps instead of raw images. The feature maps contain both spatial information and local semantics, which helps the Discriminator to more easily build the correspondence between real and fake inputs.

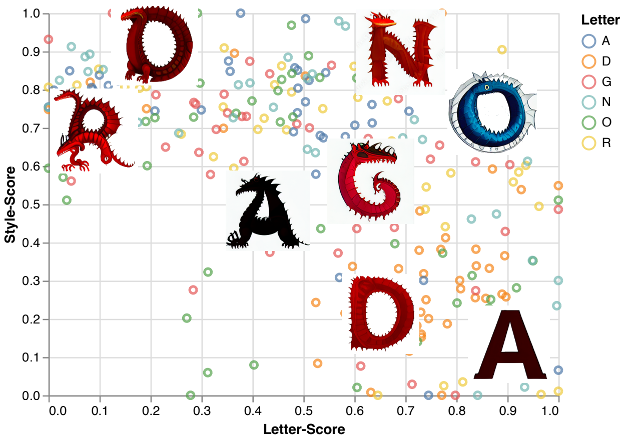

Figure 3. Ranking results. The horizontal and vertical axes respectively denote the scores of glyph and stylistic preservation.

Specifically, the rasterized glyph images are fed into the encoder to obtain the glyph feature maps z_g. Afterwards, the feature maps of style and glyph images ($\hat{z}_s$ and z_g) are sent into a CNN-based discriminator as fake and real samples, respectively. The discriminator loss $\mathcal{L}_{dis}$ is Binary Cross-Entropy Loss of real/fake prediction:

$$\mathcal{L}_{dis} = \log(D(z_g)) + \log(1 - D(\hat{z}_s)),$$

where $D(\cdot)$ denotes the function of the discriminator. Adversarially training $\mathcal{L}_{dis}$ will fine-tune the denoising generator to adapt the prompt to result incorporating the style of input images but following the shape of glyph.

3.4. Overall loss function and result ranking

The overall loss function is composed of the discrimination loss and the diffusion loss. We alternately optimize the Discriminator D and the Denoising Generator G:

$$\min_G \max_D (\mathcal{L}_{diff} + \lambda \mathcal{L}_{dis}), \tag{1}$$

where λ is a hyperparameter that makes a tradeoff between maintaining the shape of letters and incorporating characteristics of style images. Typically, λ works best set less than 1 and is experimentally set to 0.01. The effect of different λ values on results and training is shown in ablation. The combination of diffusion loss and discriminator loss results in images that incorporate elements of the style while maintaining a structure similar to the glyph.

Since our model outputs several candidate images from random noises, we have designed a strategy to select better candidates automatically. We employ CLIP to judge the quality of results from both stylistic and glyph preservation standards. Two different prompts are employed to judge each individually: the first is the glyph content like "Letter R" and the second is style word like "Dragon". Figure 3 shows a visual example of ranking a set of stylized glyphs, where the most top-right results are preferred.

4. Results and evaluation

To evaluate our DS-Fusion for artistic typography generation, we present qualitative results with single-letter and whole-word inputs and show the effects of input fonts, style attributes, and single- vs. multi-font training. Qualitative and quantitative comparisons are made to the closest alternatives that can be adapted to produce similar forms of typographies, including DALL-E 2 [20], Stable Diffusion (SD) [21], CLIPDraw [8], and a Google search baseline. As artistic creations are often judged subjectively, we conduct user studies for our comparative studies including those against searchable contents produced by human artists.

Inputs. We test our method on input style prompts and glyphs that are applicable to a variety of object categories such as those from animals, plants, professions, etc. The input choices reflect an intention to generate compelling artistic typographies, as well as to facilitate comparisons to baseline approaches and artist creations. Particularly, where possible, we attempt to produce results using DS-Fusion on inputs picked in other works to ensure fairness and stress test our method. On the other hand, the input fonts are randomly selected from Ubuntu built-in font library.

Training details. For all experiments, we use the pre-trained Latent Diffusion Model with BERT [21], An Adam optimizer with learning rate of 1e-5 is applied for the denoising generator and for the discriminator, the learning rate is 1e-4. All the hyper parameters associated with the Latent Model are kept to their default values. The complete network, including the discriminator, is fine-tuned to one particular style and glyph combination. We train the network for 800 epochs for one-font mode and 1,200 epochs for multi-font mode. On a NVIDIA GeForce RTX 3090, this takes ∼5 minutes and ∼8 minutes, respectively, to obtain the final results.

Parameters. There are two free parameters in our method: the number of style images and discriminator weight λ. Both were set experimentally and fixed throughout our experiments. We provide an ablation study in Section 4.4.

Evaluation metrics. Objectively, we evaluate the generated typographies for their legibility via optical character recognition provided by EasyOCR [12] and style incorporation via CLIP. Specifically, we use affinity scores calculated by CLIP between the generated typography results and those produced with the prompt "Picture of <style word>." Subjective tests to additionally assess the artistry and creativity of the results are conducted via user studies.

4.1. Qualitative results

Single-letter input. In Figures 1 and 4, we show a sampler of compelling single-letter artistic typography results generated by our method. Each result is produced by composing generated single-letter results with renderings of the remaining letters in the input style word. Since the generated results may have noisy backgrounds, we employ DeepLab [5] for object segmentation to remove them. To render the other letters, we use the same font as the stylized glyph and select a complementing color which is often the dominant color in the stylized glyph. More results can be found in the supplementary material.

Multi-letter input. Synthesizing artistic typographies from multiple glyphs is more challenging due to the added structural complexities in the inputs, as well as the more global context to account for. With a higher degree of freedom in the generative process, the tradeoff between legibility and artistry becomes harder to control. In Figure 5, we show a sampler of results from *whole-word* stylization.

In more ways than one, our DS-Fusion demonstrates its ability to utilize all the letters of a word to convey semantic features, in a creative manner. Particularly compelling are the two results for the style word "OWL"; the owls' bodies are well formed by the stylized glyphs. Auxiliary stylizations can also be generated to offer a global context, e.g., the tree and mountain tops, while the ship image shows the word "SHIP" replacing the vessel itself, even including a reflection in water. On the other hand, stylization could also be applied to individual letters, e.g., for "PARIS."

Effect of input fonts. Figure 6 shows results produced by varying the fonts of the input glyphs, for "dragon" + $\mathscr{G}$(A). They highlight the ability of our method to preserve various shape characteristics of the input, such as stroke thickness, slanting, and even small-scale details such as the accent on top of the letter stylized as dragon heads.

Effect of style attributes. The use of style attributes in the input prompts can be an effective means to further fine-tune the stylization, as shown in Figure 7.

Single- vs. multiple-font training. In single-font mode, we pick one random font out of a set of five selected fonts. During training, we change the color of the font randomly. Results of the single-font training show a high degree of shape correlation to the input font, as shown in Figure 6.

In multiple-font mode, we pick both a random font and a random color in each training step. It takes longer for the model to converge, but it can produce more abstract results, as reflected by the comparisons in Figure 9. Note that we

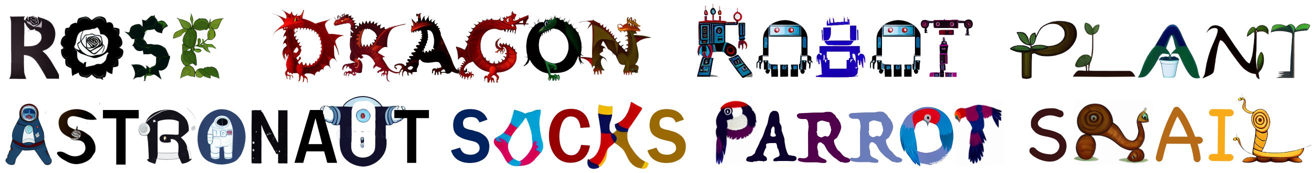

Figure 4. Single-letter artistic typography generated by DS-Fusion, demonstrating the quality and versatility of the stylization based on semantics of the word and the input glyph shapes. First row: all letters are stylized. Second row: selected letters are stylized.

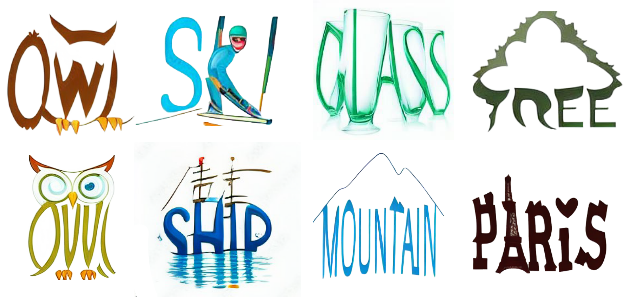

Figure 5. Results from DS-Fusion to stylize whole words, showing artistic letter combination and auxiliary complementation.

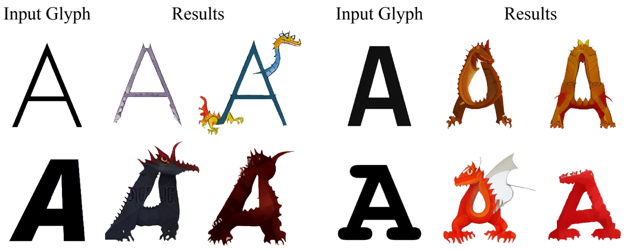

Figure 6. Details in the input fonts, e.g., thickness, slanting, and accents, are well reflected in results produced by our method.

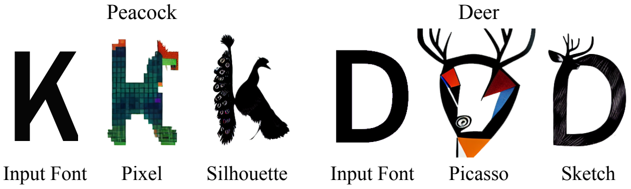

Figure 7. Additional style attributes such as "pixel", "sketch", and "Picasso" can be well reflected in our results.

Method	OCR↑	OCR [blurred]↑	CLIP↑
DALL-E 2 [20]	4.9	9.8	22.3
SD [21]	7.0	4.9	**23.1**
CLIPDraw [8] (SF)	4.6	15.8	22.7
DS-Fusion (SF)	**36.4**	**61.8**	21.1
CLIPDraw [8] (MF)	**6.4**	17.9	23.2
DS-Fusion (MF)	5.1	**20.5**	**24.1**

Table 1. Quantitative comparisons between different methods in terms of legibility (OCR) and style incorporation (CLIP). "SF" and "MF" denote single- and multi-font inputs, respectively.

cannot control the font shape in this mode, however, it follows the general shape of the input glyph.

4.2. Comparisons

Qualitative comparisons. We compare DS-Fusion to DALL-E 2 [20], vanilla Stable Diffusion (SD) [21], Google Search, and CLIPDraw [8] for which we modify its input from random strokes to vector outlines of the input glyphs to fit the task at hand. As we can see from the visual comparisons in Figure 8, in most cases, DALL-E 2 cannot blend the semantics into the letters; the objects simply surround the letters. Results from SD usually do not contain the shape of the input letters. Google search could occasionally return good results only because the input prompts have had corresponding artist designs that were retrieved. For less common queries such as "Snail in letter N," retrieval clearly cannot work. As for CLIPDraw, neither the content legibility nor the semantic embedding is satisfactory.

Quantitative comparisons. To evaluate the performance of different methods in terms of legibility and style incorporation, we tested on 21 style words across a wide range of semantic categories (see supplementary material for the list) and stylized all the letters from them. Since both CLIPDraw and our method can receive multiple fonts as input, we also separately report their performances when receiving single and multiple fonts. For each style word, 4 *random* results were sampled from each method to obtain statistics.

The quantitative results are shown in Table 1. On single-font inputs, our method significantly outperforms the others in OCR accuracy both for raw and blurred versions of the generated results. The style score of DS-Fusion is slightly behind the others, which is not unexpected since in one-font mode, the output favors the glyph, and the style is less pronounced; see Figure 9. Since CLIPDraw is over-fitted to the CLIP loss, its style score tends to be high while the corresponding visual results are not satisfactory; see Figure 8 and the following user study. When taking multiple fonts as input, DS-Fusion achieves the best score (24.1) of style incorporation among all methods. The letters are stylized in a much more abstract fashion, which also makes them more difficult to be recognized. Nevertheless, the OCR accuracy of DS-Fusion is comparable to CLIPDraw.

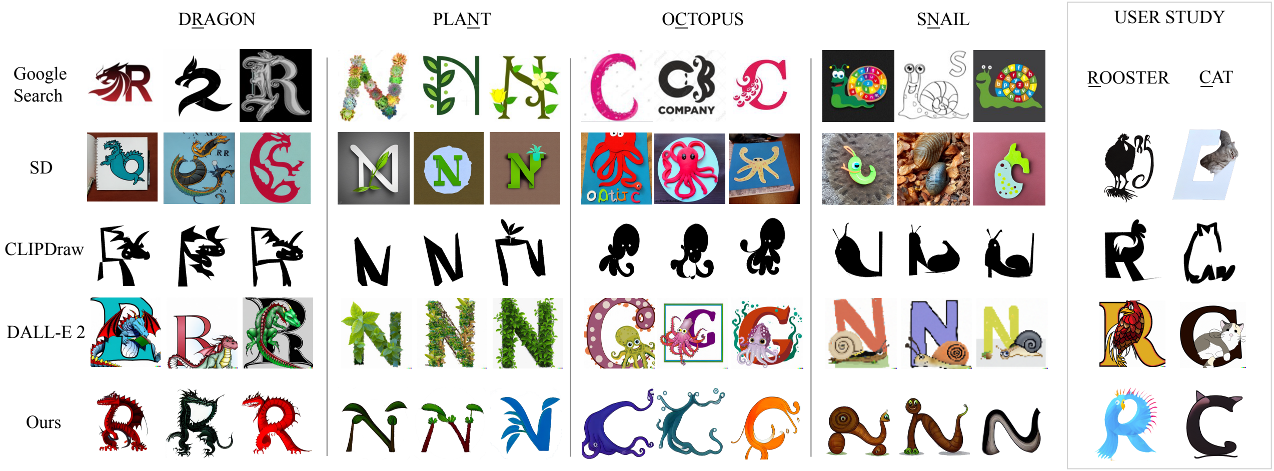

Figure 8. Visual comparison on single-letter (underlined) generation between different methods, with three most compelling results picked per method. The last two columns show sample results from the first user study (Section 4.3), where Google search was not considered.

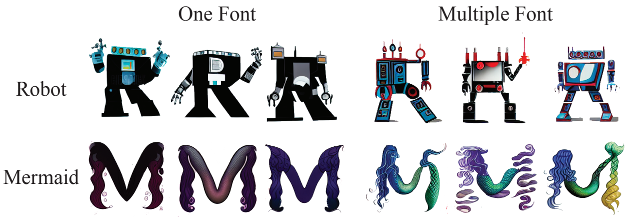

Figure 9. One-font vs. multi-font training. Results show a stronger legibility in the former, and a more abstract stylization in the latter.

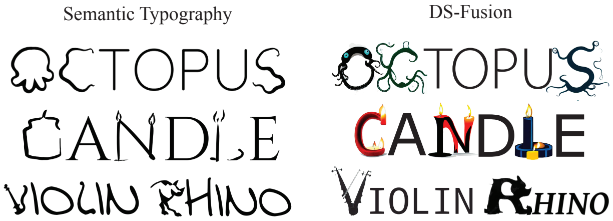

Figure 10. Visual comparison with semantic typography [10].

Comparison with Semantic Typography. We compare DS-Fusion to Word-as-Image Semantic Typography [10] in Figure 10. both methods perform similarly; both results are still readable and semantically relevant (e.g., rhino, violin). However, our method is capable of adding colorful and artistic textures to make the results more semantically relevant and visually appealing, e.g., candle and octopus.

4.3. User Study

A user study serves to evaluate our method via subjective human judgement. We perform two such studies: in the first one, users select between DS-Fusion and other generative methods; in the second, a different group of participants selects between DS-Fusion and artist designs. Both studies start by showing participants the definition and examples of artistic typography, which give them an idea of what to expect. We directly chose *artist-created* examples from a popular online tutorial on artistic typography design, instead of picking inputs that may favour our method. The same 10 inputs with distinguishable style words (e.g.,Cat, Parrot, etc.) were used in both studies.

The first study compares our results to those from CLIPDraw, DALL-E 2, and SD. To prepare the study, we asked a professional designer to select the most representative result for each of the four candidate results generated by each method. The study collected responses from 32 participants with different occupations and varying artistic backgrounds to determine which result was deemed best. Table 2 shows that DS-Fusion significantly outperformed other methods.

While DALL-E 2 also gathered close to 30% votes, some of its results preferred by users do not contain any stylization of the input letters, e.g., see the last two columns of Figure 8. DALL-E 2 simply placed an image of a cat or rooster near an un-stylized letter. Clearly, these are not satisfactory typography results, yet user subjectivity has led to them receiving 25% and **50%** of the votes, respectively.

The second study collected responses from 42 participants to choose between our results and professionally crafted examples found in the tutorial. The study results are shown in the second part in Table 2. In about 42% of the cases, users found our results better or equivalent to that human-designed examples. This is a satisfactory outcome since human-designed examples heavily rely on the creativity and expertise of designers (more details about our user study in the supplementary material).

4.4. Ablation studies

Discriminator weight. The impact of the λ value on adjusting the Discriminator loss is shown in Figure 11. A high

CLIPDraw	DALL-E 2	SD	Equal	DS-Fusion
6.88	29.38	9.38	5.31	**49.06**

Human	Equal	DS-Fusion
57.14	11.19	31.67

Table 2. User Study comparing DS-Fusion with other generative methods (Top) and human designs (Bottom). The numbers show the percentage of user preference for different results. "SD" denotes Stable Diffusion. "Equal" denotes equally good.

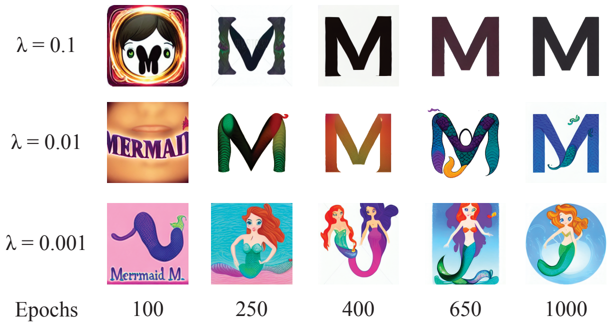

Figure 11. Effect of discriminator weight, on "Mermaid" + $\mathscr{G}$(M).

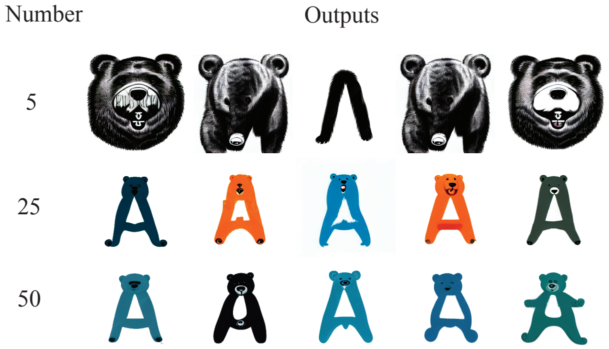

Figure 12. Effect of generating a different number of style images. The left column shows the number of images generated, and on the right, we see the range of output results.

value of λ pushes the generator towards the glyph shape, However, the generator may take a long time to converge to the glyph shape or may not be able to reach it at all when a small value of lambda is used. Therefore, the value of lambda was selected experimentally.

Number of style images. In Figure 12 we show the effect of generating different numbers of style images. When the number is too low, the method struggles to extract common features to apply to the glyph. As the number of images increases, we see not only improvement in generation quality but also in the diversity of outputs. We do not observe a significant improvement in increasing the value excessively, hence we opt for the number 25 as an optimal balance.

Text encoder. Our results are generated using BERT. This model has a smaller resolution and is faster to generate compared to CLIP SD. However, we show some results using the CLIP text encoder in Figure 13 to demonstrate the generality of our method.

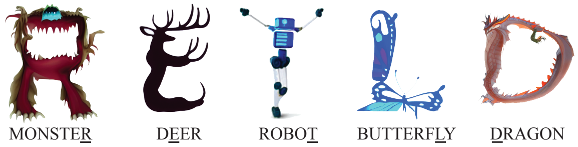

Figure 13. Results on using CLIP encoder for text conditioning.

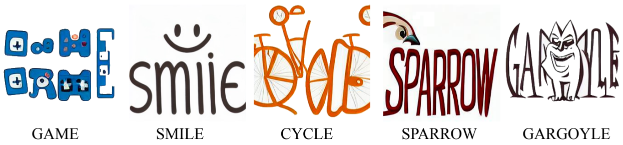

Figure 14. Some failure cases on whole-word inputs.

5. Conclusion, limitation, and future work

We developed an automatic method to generate artistic typography for a letter or word by using language-based generative models equipped with a discriminator. We demonstrated the effectiveness of our method, coined DS-Fusion, through extensive experiments and user studies. Our approach combines adversarial learning and diffusion, which helps ensure the fidelity of the generated typography.

Our method is generally effective but it still has some limitations that require further investigation. When dealing with multi-letter inputs, our method may struggle to generate satisfactory results if the style images and letters are too dissimilar (Figure 14). At present, our method is optimized for each specific combination of style and glyph. However, future work could involve training a network for a particular style that can generate any letter during inference. Despite having an automatic selection strategy, it may not always generate the most visually plausible outcome. Rather, it can generate a range of plausible results to choose from. A stronger selection mechanism could be a future work.

Our glyph is potentially an image in various forms, including alphanumeric fonts, foreign language characters, or even a 2D shape. It is interesting to creatively modify such shapes to display a semantic (e.g., a flower-shaped chair). In addition, for personalization, style images can be manually prepared or drawn if desired. We explored these ideas and presented preliminary results in the supplementary material, but further research is necessary to solidify the outcomes.

References

[1] Samaneh Azadi, Matthew Fisher, Vladimir G 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*, pages 7564–7573, 2018. 2

[2] Elena Balashova, Amit H Bermano, Vladimir G Kim, Stephen DiVerdi, Aaron Hertzmann, and Thomas Funkhouser. Learning a stroke-based representation for fonts. In *Computer Graphics Forum*, volume 38, pages 429–442. Wiley Online Library, 2019. 2

[3] Daniel Berio, Frederic Fol Leymarie, Paul Asente, and Jose Echevarria. Strokestyles: Stroke-based segmentation and stylization of fonts. *ACM Transactions on Graphics (TOG)*, 41(3):1–21, 2022. 1, 2, 3

[4] Neill DF Campbell and Jan Kautz. Learning a manifold of fonts. *ACM Transactions on Graphics (TOG)*, 33(4):1–11, 2014. 2

[5] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017. 5

[6] Minghai Chen, Fan Xu, and Lin Lu. Manufacturable pattern collage along a boundary. *Computational Visual Media*, 5:293–302, 2019. 3

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019. 3

[8] Kevin Frans, Lisa B Soros, and Olaf Witkowski. Clipdraw: Exploring text-to-drawing synthesis through language-image encoders. *arXiv preprint arXiv:2106.14843*. 3, 5, 6

[9] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020. 3

[10] Shir Iluz, Yael Vinker, Amir Hertz, Daniel Berio, Daniel Cohen-Or, and Ariel Shamir. Word-as-image for semantic typography. *arXiv preprint arXiv:2303.01818*, 2023. 1, 3, 7

[11] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017. 2

[12] AI Jaided. Easyocr. *Retrieved October*, 9:2020, 2020. 5

[13] Ajay Jain, Ben Mildenhall, Jonathan T Barron, Pieter Abbeel, and Ben Poole. Zero-shot text-guided object generation with dream fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 867–876, 2022. 3

[14] Kin Chung Kwan, Lok Tsun Sinn, Chu Han, Tien-Tsin Wong, and Chi-Wing Fu. Pyramid of arclength descriptor for generating collage of shapes. *ACM Trans. Graph.*, 35(6):229–1, 2016. 3

[15] Tzu-Mao Li, Michal Lukáč, Michaël Gharbi, and Jonathan Ragan-Kelley. Differentiable vector graphics rasterization for editing and learning. *ACM Transactions on Graphics (TOG)*, 39(6):1–15, 2020. 3

[16] Oscar Michel, Roi Bar-On, Richard Liu, Sagie Benaim, and Rana Hanocka. Text2mesh: Text-driven neural stylization for meshes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13492–13502, 2022. 3

[17] B Mildenhall, PP Srinivasan, M Tancik, JT Barron, R Ramamoorthi, and R Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision*, 2020. 3

[18] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 2

[19] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 1, 2, 3

[20] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022. 2, 5, 6

[21] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022. 1, 2, 3, 4, 5, 6

[22] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015. 4

[23] Reza Adhitya Saputra, Craig S Kaplan, and Paul Asente. Improved deformation-driven element packing with repulsionpak. *IEEE transactions on visualization and computer graphics*, 27(4):2396–2408, 2019. 3

[24] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2256–2265, Lille, France, 07–09 Jul 2015. PMLR. 3

[25] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv:2010.02502*, October 2020. 3

[26] Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA, 2020. Curran Associates Inc. 3

[27] Purva Tendulkar, Kalpesh Krishna, Ramprasaath R Selvaraju, and Devi Parikh. Trick or treat: Thematic

reinforcement for artistic typography. *arXiv preprint arXiv:1903.07820*, 2019. 1, 3

[28] Can Wang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. Clip-nerf: Text-and-image driven manipulation of neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3835–3844, 2022. 3

[29] Yizhi Wang and Zhouhui Lian. Deepvecfont: synthesizing high-quality vector fonts via dual-modality learning. *ACM Transactions on Graphics (TOG)*, 40(6):1–15, 2021. 2

[30] Zhendong Wang, Huangjie Zheng, Pengcheng He, Weizhu Chen, and Mingyuan Zhou. Diffusion-gan: Training gans with diffusion. *arXiv preprint arXiv:2206.02262*, 2022. 3

[31] 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. 2

[32] Shuai Yang, Jiaying Liu, Wenhan Yang, and Zongming Guo. Context-aware text-based binary image stylization and synthesis. *IEEE Transactions on Image Processing*, 28(2):952–964, 2018. 2

[33] Shuai Yang, Jiaying Liu, Wenhan Yang, and Zongming Guo. Context-aware unsupervised text stylization. In *Proceedings of the 26th ACM international conference on Multimedia*, pages 1688–1696, 2018. 2

[34] Junsong Zhang, Yu Wang, Weiyi Xiao, and Zhenshan Luo. Synthesizing ornamental typefaces. In *Computer Graphics Forum*, volume 36, pages 64–75. Wiley Online Library, 2017. 1, 3

[35] Junsong Zhang, Zuyi Yang, Linchengyu Jin, Zhitang Lu, and Jinhui Yu. Creating word paintings jointly considering semantics, attention, and aesthetics. *ACM Transactions on Applied Perceptions (TAP)*, 19(3):1–21, 2022. 3

[36] Changqing Zou, Junjie Cao, Warunika Ranaweera, Ibraheem Alhashim, Ping Tan, Alla Sheffer, and Hao Zhang. Legible compact calligrams. *ACM Transactions on Graphics (TOG)*, 35(4):1–12, 2016. 3