Multimodal Generative Model for Image Manipulation

Harsh Sunil Lakhani
The Hong Kong University of Science and Technology
hslakhani@connect.ust.hk

Li Qi
The Hong Kong University of Science and Technology
qlicm@connect.ust.hk

Abstract

In this project, we implement and compare two popular generative models, styleCLIP and latent diffusion model, to perform image to image manipulation with a text description as an input to guide the modification to be made. Most of the existing generative models only take one type of input, which is either image or text. We attempt to build a generative model that takes text and image simultaneously. After conducting a comprehensive survey, we find an elegant model CLIP, which enables us use two encoders that is capable of encoding both, the image and the text input to a latent space with coherent representations, so that the generative network could produce a satisfying result.

1. Introduction

Generative models is one of the main topics in the computer vision realm. With the development of Convolutional Neural Networks(CNN) and Vision Transformers(ViT) [5], a lot of generative models are proposed, including VAE [12], GAN [7], styleGAN [9] and the latest diffusion models, which generate realistic images in accordance to customized inputs from a user. Most generative models are a single-input model, i.e. allow only one type of input from either image or text. It is known that the diffusion model is a multi-modal model that allows for both, image and text as inputs simultaneously and generates an output in accordance to both of the inputs. However, as we demonstrate later, the generated result is not satisfying especially when it comes to manga-style images for a pretrained diffusion model that is not trained on a manga image dataset. In this project, we make use of a robust data set containing a huge amount of manga-style images to train our own diffusion model that is able to manipulate a given image in accordance to the guidance provided by the input text.

2. Related Work

2.1. Generative Adversarial Networks (GANs)

The use of GANs [7] has increased severely in the past few years due to the increase in the ability to access higher computational resources. GANs have been used in several use cases and has achieved impressive results. The most notable use cases include image generation, representation learning and image editing. Recently, GANs have been used to work on more complex methods including text2image [15], future prediction and image inpainting. The success of GANs is attributed to the feature of adversarial loss, which forces the generated images to be impossible to differentiate from real images. Zhu *et al*. [18] adopted an adversarial loss to help learn the mapping which in turn causes the translated image to not be differentiable from the target domain's images.

2.2. Diffusion Probabilistic Model

Diffusion models [8] fall under the class of likelihood-based models. In the recent years, these likelihood-based models have shown to produce high-quality images. Moreover, they offer properties including distribution coverage, a stationary training objective as well as easy scalability. Essentially, these models generate samples by gradually taking out the noise from a signal. The training objective can also be looked as a re-weighted variation. These models still lag behind GANs on difficult generation data sets such as LSUN and ImageNet. Dhariwal *et al*. [4] hypothesize that this gap exists because the model architectures used by GANs have been refined and explored thoroughly. They also believe another reason behind the gap could be because of the fact that GANs are able to trade off diversity for fidelity which results in high quality samples produced. However, it does not cover the whole distribution.

2.3. Vision Language Joint Representation

Contrastive Language-Image Pre-training (CLIP) [14] model is trained on 400 million text-image pairs. CLIP is built on the work done on zero-shot transfer, natural lan-

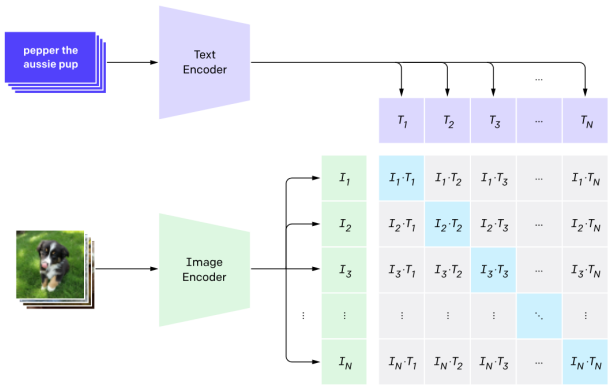

Figure 1. CLIP Encoder

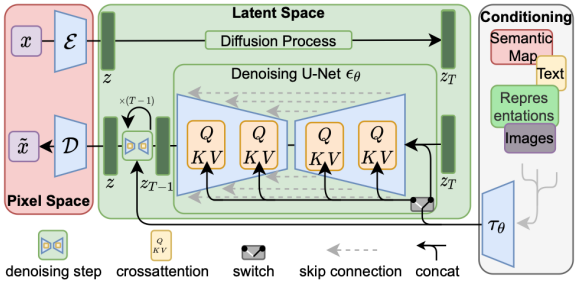

Figure 2. Latent Diffusion Model Architecture

guage supervision , and multimodal learning. The representations learned by the CLIP model helps to enable state of the art zero shot image classification on numerous data sets. CLIP models learn a wide variety of visual concepts from natural language and therefore are more flexible and general than existing models. Moreover, CLIP is highly efficient since it learns from unfiltered, raw, highly varied, and highly noisy data. Generally, such models working with this kind of data requires significant computational power, however, the OpenAI [1] team has worked on methods to reduce the required compute, which include, adopting of a contrastive objective for connecting text with images, and adopting the Vision Transformer [5].

3. Methodology

3.1. CLIP

CLIP originally was proposed to help efficiently learn visual concepts with natural language supervision [11]. It is a dual language-image encoder [17] that contributes to a large step towards unifying visual and textual information. More specifically, the CLIP model consists of two sub-models called encoders. A text encoder that embeds the text and an image encoder that embeds the images. The images and text are mapped onto the same representational space, hence, it enables us to quantify and measure the similarity between the textual descriptions and images.

A pretrained CLIP model c along with paired training data for the diffusion model, $(\mathbf{x}, y)$, where x is an image and y is the corresponding caption, allows us to compute the CLIP image and text embedding, $\mathbf{c}^t(\mathbf{x})$ and $\mathbf{c}^t(y)$, respectively.

The method to calculate the similarity between the embeddings is by making use of cosine similarity. As shown in 1, the light blue squares represent where the text and the image embedding coincide.

Many standard models are capable of knowing only very specific features and objects that they are trained on, this is where CLIP can offer generalizability, where it is able to expand the knowledge of classification models to a wider array features and objects by leveraging semantic information in text [3]. CLIP works by understanding the meaning of the classes compared to a standard model that discards the semantic meaning of the class labels. Moreover, CLIP is built on images and captions that already exist, which makes it easier to implement and doesn't require additional human time spent labeling. In addition to these advantages, CLIP is comparatively faster.

In this project, as mentioned in the sections followed, we analyze and compare performance of styleCLIP based on the pretrained styleGAN2 architecture, and the performance of the latent diffusion model.

3.2. Latent Diffusion Model

Diffusion Models as mentioned earlier have shown to produce high-quality images while offering properties including distribution coverage and easy scalability. The best quality is usually achieve when a reweighted objective us used for training. However, this is where diffusion models corresponds to a lossy compressor and allows to trade quality for compression ability. It is known that evaluation and optimization of these models in pixel space is highly expensive and slow. The speed can be improved by advanced sample strategies, however, it still remains expensive to calculate gradients for training on high-resolution images. This is where Latent Diffusion Models [16] come into play, they work on a compressed latent space of lower dimensionality which helps to make training computationally cheaper and also speeds up inference with negligible reduction in synthesis quality.

$$L_{LDM} := E_{\epsilon(x),\varepsilon\sim N(0,1),t}[\|\varepsilon - \varepsilon_\theta(z_t, t)\|_2^2] \quad (1)$$

where z_t is the latent representation, E is the encoder, $p(z)$ is the distribution, and D is the decoder.

The latent diffusion model makes use of the above mentioned loss function. Since the forward process is fixed, z_t

can be efficiently obtained from the E during training, and samples from $p(z)$ can be decoded to image space with a single pass through D.

3.2.1 AutoEncoder

Although diffusion models allow to ignore intuitively unnecessary details via under-sampling the corresponding loss terms, they still make use of a costly evaluation function in the pixel space, which requires high computation time and energy consumption. To overcome this problem, Rombach *et al*. [16] introduced an explicit separation of the compressive from the generative learning phase. To achieve this, they used an autoencoding model which learns a space that is intuitively equivalent to the image space, but in turn, reduces computational complexity.

This method serves several advantages. It helps to obtain diffusion models which are more computationally efficient since we leave the high-dimensional image space. Moreover, the latent space of the obtained compression models are used to train multiple generative models. The latent space can also be used for other downstream applications including CLIP-guided synthesis [6].

Specifically an encoder ϵ is used to compress the input image to a smaller 2D latent vector $\mathbf{z} = \epsilon(\mathbf{x}) \in \Re^{hxwxc}$, where the downsampling rate $f = H/h = W/w = 2^m, m \in N$. After which, the decoder D reconstructs the images from the latent vector, $\tilde{\mathbf{x}} = D(\mathbf{z})$. As shown in 1, the denoising and diffusion processes happen on the latent vector $\mathbf{z}$. The denoising model is able to handle flexible conditioning information for image generation as it is designed as a time conditioned U-Net and is modified with the cross-attention mechanism.

3.2.2 Diffusion Model

Diffusion models are a type of latent variable model that consists of a forward diffusion process and a reverse diffusion process. The forward process follows a Markov chain as noise is added to the data when sampling the latent variables x_t for $t = 1, ..., T$. The resulting latent variable is expressed as:

$$\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_0 + (1\alpha_t)\mathbf{w}, \mathbf{w} \sim N(\mathbf{0}, \mathbf{I}) \tag{2}$$

where $\alpha_t := \prod_{s=1}^{t}(1 - \beta_s)$. The reverse process $q(\mathbf{x}_{t1}|\mathbf{x}_t)$ is parameterized by another Gaussian transition $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) := N(\mathbf{x}_{t-1}; \mu(\mathbf{x}_t, t), \sigma(\mathbf{x}_t, t)\mathbf{I})$. $\mu(\mathbf{x}_t, t)$ can be decomposed into the linear combination of x_t and a noise approximation model $\varepsilon_\theta(\mathbf{x}_t, t)$, which can be learned by solving the optimization problem as follows:

$$\min_\theta E_{x_0 \sim q(x_0), w \sim N(\mathbf{0}, \mathbf{I}), t} \|\mathbf{w} - \epsilon_\theta(\mathbf{x}_t, t)\|_2^2 (3)$$

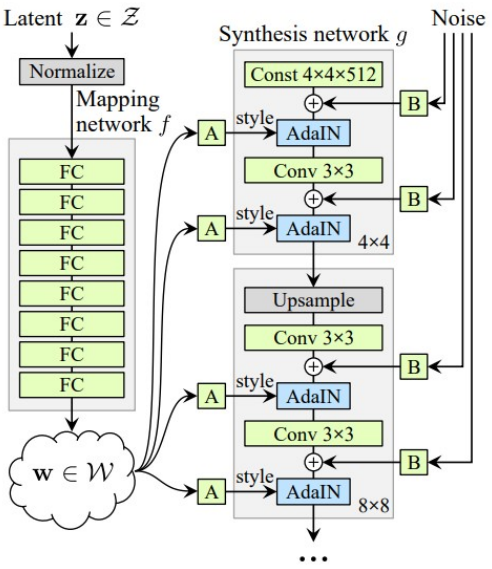

Figure 3. StyleGAN Architecture

After training $\varepsilon_\theta(\mathbf{x}, t)$, the data is sampled using following reverse diffusion process:

$$x_t = \frac{1}{\sqrt{1-\beta_t}}(x_t - \frac{\beta_t}{\sqrt{1\alpha_t}}\epsilon_\theta(\mathbf{x}_t, t)) + \sigma_t \mathbf{z}, \tag{4}$$

where $z \sim N(\mathbf{0}, \mathbf{I})$.

Diffusion models have depicted strong performance as generative models, more specifically for it's application to inverse problems, where they have demonstrated state-of-the-art performance. However, diffusion models have their limitations when we consider that they are inherently slow to sample from and need thousand iteration steps to generate images from pure Gaussian noise.

3.3. StyleCLIP

Besides diffusion model, StyleCLIP [13] is another main stream model that can do image modification with the guidance of a text input, however, the training cost is much lower compared to the diffusion model.

3.3.1 StyleGAN and its improvement

StyleGAN is a style-based generator which enables control over image synthesis. As shown in 3, Unlike a traditional GAN which takes the latent vector sampled from Gaussian distribution as input and feed it into the generative network directly, StyleGAN uses a mapping network to affine the latent vector $z \in Z$ to an intermediate latent vector $w \in W$. The purpose of this mapping network is to try and disentangle the latent vector z so that different features in the gen-

erated image can be controlled independently by modifying the intermediate latent vector w in a correct manner.

Different from a conventional GAN where the latent vector z only contributes to the generation at the very beginning, when fed into the synthesis network, the intermediate latent vector controls the whole generative process after affined to a style vector by adaptive instance normalization(AdaIN) at each convolution layer. The normalization is done within each instance rather than in batches as seen in AdaIN. The operation in AdaIN can be expressed as following equation:

$$AdaIN(X_i, y) = y_{s,i}\frac{x_i - \mu(x_i)}{\sigma(x_i} + y_{b,i} \tag{5}$$

where x_i is the i-th feature map, $y_{s,i}$ is the instance's scale factor, and $y_{b,i}$ is the translation factor. To facilitate more stochastic variation in the generated image, Gaussian noise is also added to every convolution layer. The synthesis network takes a learned constant vector as input, and with the help of the above mentioned operations, it generates an image with the desired resolution.

The styleGAN has made great success in image generation and enables control over it as well as style mixing. However, it suffers from the problem of characteristic artifacts in generated images. To solve this problem, the author revised the architecture of styleGAN in following ways mentioned. First, remove the redundant operations to the constant input at the beginning. Second, move the addition of b and random noise B to be outside active area of a style. Third, adjust only the standard deviation rather than both, the mean and standard deviation per feature map. By doing this, the architecture is changed from b to c in 3, which is known as styleGAN2. [10] In addition, the author replaces instance normalization in the revised architecture with a "demodulation" operation, which will not be introduced in detail here. Our implementation of styleCLIP is based on a pretrained styleGAN2 architecture described here on the FFHQ data set.

3.3.2 Latent Space manipulation with StyelCLIP

Patashnik et al [13] proposed three ways for text-driven image manipulation, all combining the joint image-text embedding learned by CLIP and the generative power of styleGAN.

The first way is called latent optimization. Given a source latent code w_s, it directly optimizes the latent code w in W^+ space with the following objective:

$$\arg\min_{w \in \mathcal{W}+} D_{CLIP}(G(w), t) + \lambda_{L2}||w - w_s||_2 + \lambda_{ID}\mathcal{L}_{ID}(w) \tag{6}$$

where $D_{CLIP}(G(w), t)$ represents the cosine distance between the generated image controlled by latent code w and given prompt t, $\lambda_{L2}||w - w_s||_2$ and $\lambda_{ID}\mathcal{L}_{ID}(w)$ ensure that the generated image is still close to the original image. $\mathcal{L}_{ID}(w)$ represents the ID loss calculated by a pretrained face recognition network.

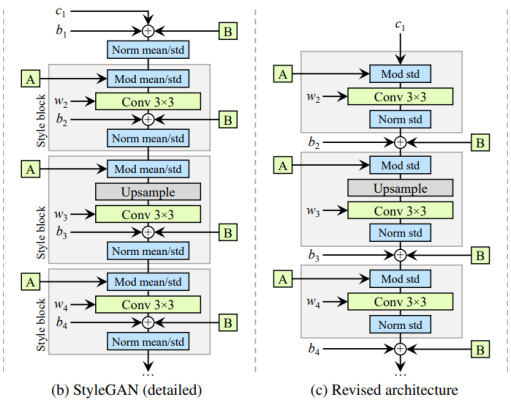

Figure 4. Revised StyleGAN2 Architecture

The latent optimization is intuitive and easy to implement. However, it requires several minutes to optimize the latent code for every (image, prompt) pair, which is not efficient and does not provide real-time inference. To overcome this problem, another method called latent mapping is proposed, wherein we are required to train a mapping network to map the latent code w^+ in W^+ space to the manipulated one which matches the prompt for several hours per text prompt. Once training is completed, the mapper network can be used to manipulate all the images with this prompt in almost real time.

The architecture of mapper network is shown in 5, the mapper network consist of three simple fully connected networks, which has similar architecture to that of the styleGAN mapping network but has fewer layers in comparison. Each sub mapping network is responsible for mapping different parts of the latent vector, as they are used to control different features when generating the image. The mapper network can defined by the following equation

$$M_t(w) = (M_t^c(w_c, M_t^m(w_m), M_t^f(w_f) \tag{7}$$

and similar to the loss function of latent optimization, the mapper network can be trained by minimizing this loss to optimize the parameters of the network rather than the latent code

$$\mathcal{L}(w) = \mathcal{L}_{CLIP}(w) + \lambda_{L2}||M_t(w)||_2 + \lambda_{ID}\mathcal{L}_{ID}(w) \tag{8}$$

where $\mathcal{L}_{CLIP}(w)$ is the cosine distance between the image generated with w and the text prompt in CLIP embedded space, and $\lambda_{L2}||M_t(w)||_2$ makes sure that the mapped offset, which will be added to original latent vector, is small so that the remaining parts of generated image, which are irrelevant to the text prompt, remain unchanged. $\lambda_{ID}\mathcal{L}_{ID}(w)$

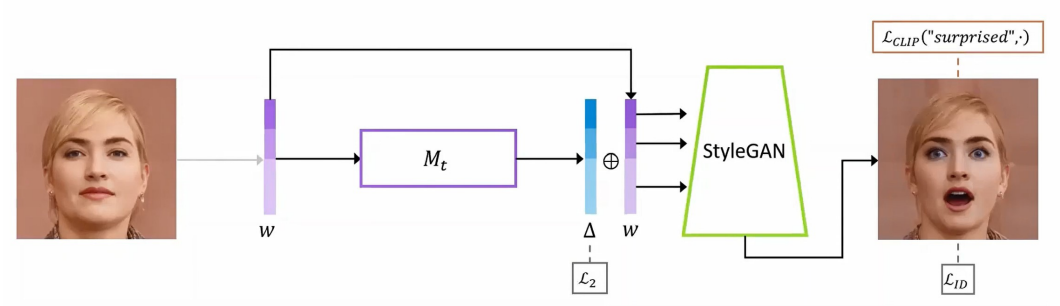

Figure 5. Architecture for Latent Mapper

(a) prompt: A person with purple hair

(b) prompt: surprised

Figure 6. Comparison of results generated by diffusion model and styleCLIP model

(a) prompt: blue eyes

Figure 7. Result generated by pretrained diffusion model on a manga image with blue eyes as prompt

serves a similar purpose to that of the loss of latent optimization, which is used for reserving the characteristic of the original image.

The last scheme for image manipulation proposed by styleCLIP is global direction. It transforms a given text prompt into an input agnostic mapping direction in styleGAN's style space S. The details of global direction is hard to be described in a concise manner, and as it is not the main focus of our project, we will not introduce it in detail.

4. Experiments

4.1. Preliminary Experiments

As training a diffusion model or styleGAN2 is very expensive for us, we first perform some experiments on the some pretrained models to compare their performance on human face image modification with the same input image and prompt text. Figure 6 shows the results of our preliminary experiments. It is worth pointing that the comparison is not fair since the styleCLIP is trained on a human face dataset FFHQ while stable diffusion is trained on a more diverse dataset LAION-5B.

Since our goal is to perform text guided manga image manipulation and achieve a desirable result, we first test if we can directly apply the above mentioned pretrained models on our data set containing manga-styled images. However, when attempting this, we come across a few difficulties. First, text guided image manipulation with styleCLIP requires an encoding network that can encode the source image to latent code of styleGAN, which is difficult for us to obtain. It is feasible to use the pretrained diffusion model to manipulate the manga-styled images guided by text prompt However, the result is not satisfying as shown in Figure 7. We blame this result on the domain shift problem.

After conducting the preliminary experiments, we plan to train our own latent diffusion model. The implementation details are as follows.

4.2. Data set

The data set selected to train our model is the Danbooru2017 [2] data set. Developed by Gwern Branwen and the Danbooru Community, the Danbooru2017 data set is a popular data set that is often used for rich large-scale tag-

ging and learned embedding. It is also used to test out the transfer-ability of existing computer vision techniques to anime-style images. The data set contains approximately 2.9 million 256x256 manga-style images, along with an average of 37 annotations (tags) describing the objects and features in each image. The entire data set including the images and the metadata are approximately 1.9 terabytes. For the purpose of our experiments, we only selected a subset of the entire data set which contains 300,000 images and is approximately 45 gigabytes including the metadata.

The original metadata is stored in a nested JSON format. In order to use the metadata in our model and experiments, we had to prepare and transform the data into an appropriate format. To achieve this in an efficient manner, we made use of Big Query on Google Cloud.

4.3. Experiment Setting

Following the training schema of latent diffusion model, we first trained an autoencoder-kl with scaling factor = 8, which downsamples the original 256x256x3 image to the latent space with shape of 32x32x4. After the training of the autoencoder converges, we freeze it to train the Unet for diffusion model together with a pretrained clip-vit-large-patch14 CLIP text enbedder serving as the conditional text information encoder. The hyper parameters for training the autoencoder follows the setting in the configuration of autoencoder-kl-32x32x4 provided by the author of the latent diffusion model. The hyper parameters for training the denoising Unet is based on learning different configurations provided by the author, since the configuration for training a text conditional latent diffusion model is not provided. We set the batch size to 8 due to the limitation of GPU memory, and decrement the learning rate correspondingly. We select AdamW as our optimizer, and schedule a warm-up for 10000 steps at the beginning of training.

4.4. Experiment Results

Here are some examples generated by our trained model in figure 8, compared to the pretrained model which is not trained on a manga data set. It can be observed that the pretrained model failed to generate high quality images and cannot manipulate the input image in accordance to the text prompt as desired, whereas, our model trained on Danbooru2017 data set is able to produce manipulated images with high quality that are consistent with the text prompt.

5. Conclusion

In this project, we first compared styleCLIP and diffusion model on human face image modification according to a text input that serves as a guidance. Initially, we were most concerned to figure out a method to build two encoders which was capable of encoding of the inputs to a latent space with coherent representations. However, after conducting a comprehensive survey, we found an elegant model, CLIP, that helped us overcome this problem. After implementing and performing analysis on the preliminary experiment results produced, we concluded that the diffusion model's performance was unsatisfactory while the performance of the styleCLIP was considerably more impressive. Therefore, we attempt to go ahead with styleCLIP but face a major problem that we cannot obtain the latent code of the source image. Hence, we decided to train our own latent diffusion model. Finally, with the latent diffusion model trained on Danbooru, we were able to produce satisfying results as desired to achieve our goal.

(a) prompt: wink

(b) prompt: pink hair

(c) prompt: wear glasses

Figure 8. Three simple graphs

References

[1] Jong Wook Kim Gretchen Krueger Sandhini Agarwal Alec Radford, Ilya Sutskever. Clip: Connecting text and images. https://openai.com/blog/clip/. 2

[2] Anonymous, Danbooru community, and Gwern Branwen. Danbooru2021: A large-scale crowdsourced and tagged anime illustration dataset. https://www.gwern.net/Danbooru2021, January 2022. Accessed: DATE. 5

[3] Matthew Brems. A Beginner's Guide to the CLIP Model. https://www.kdnuggets.com/2021/03/beginners-guide-clip-model.html. 2

[4] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021. 1

[5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner,

Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 1, 2

[6] Kevin Frans, Lisa B Soros, and Olaf Witkowski. Clipdraw: Exploring text-to-drawing synthesis through language-image encoders. *arXiv preprint arXiv:2106.14843*, 2021. 3

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020. 1

[8] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020. 1

[9] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019. 1

[10] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119, 2020. 4

[11] Gwanghyun Kim, Taesung Kwon, and Jong Chul Ye. Diffusionclip: Text-guided diffusion models for robust image manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2426–2435, 2022. 2

[12] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 1

[13] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. Styleclip: Text-driven manipulation of stylegan imagery. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2085–2094, 2021. 3, 4

[14] 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

[15] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. In *International conference on machine learning*, pages 1060–1069. PMLR, 2016. 1

[16] 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. 2, 3

[17] Lilian Weng. What are diffusion models? *lilianweng.github.io*, Jul 2021. 2

[18] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017. 1