Text-guided Image Generation with Diffusion Models and CLIP

Kao Kitichotkul
rkitichotkul@stanford.edu

Patin Inkaew
pinkaew@stanford.edu

Abstract

Text-guided image generation is a conditional generative task: producing images capturing the semantic meaning of the input text prompt. In this project, we explore the capabilities of combining publicly available pre-trained models to tackle text-guided image synthesis. Specifically, we explore recently introduced diffusion and score-based generative models, which show comparable or better performances in many generative tasks compared to Generative Adversarial Networks (GANs), but have more stable training methods. We then use Open AI's CLIP model to guide image generation. We propose two approaches: (1) Contrastive conditional diffusion sampling using one target together with multiple distractive prompts to improve sampling, and (2) CLIP-guided unconditional diffusion sampling using the gradient from CLIP-loss to nudge latent image to the desired location. We demonstrate better performance, easier and faster sampling for the first approach compared to a similar approach without contrastive prompts. The second approach shows a crucial need for an original image, rather than a completely noisy image, for successful sampling. Our project shows promising potentials in utilizing pre-trained models to generate high-quality text-guided images without training models to tackle the task specifically.

1 Introduction

In a text-guided image synthesis task, given an input text description, the model aims to synthesize an image that captures the semantic content of the text. This problem is interesting because there are cases where writing high-quality descriptions is easier than capturing high-quality images. Text-guided image generation can aid artistic creation, augment datasets with few images but rich descriptions such as medical images, and produce effective visualization.

Image synthesis is a hard problem. However, many generative models, such as variational autoencoders and generative adversarial networks (GANs), have already provided good results for unconditional image generation. Conditional image generation is a variant problem that poses constraints on the desired image output, adding another layer of complexity. In such a task, text descriptions are popular conditioning for image synthesis. Many models including classical methods and neural networks have been shown to solve text-guided image generation tasks; one notable example is Open AI's Dall-E model [1]. However, many models using GAN-based networks will suffer from mode collapse and unstable adversarial training. Moreover, a model, like Dall-E, uses a large transformer-based architecture and will require a large amount of data and training time.

We explore the capability of the recently introduced *diffusion* and *score-based generative models* to tackle text-guided image synthesis. Song et al. [2] shows both diffusion model and score-based models are equivalent. Score-based generative model estimates score $\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$ instead of directly estimate data distribution $p_{\text{data}}(\mathbf{x})$. This allows larger neural network architectures because score function $\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$ does not need to be normalized while generative models, such as normalizing flow models, need to constraint the architecture allowing partition function $\mathbf{Z}_\theta$ can be

Preprint. Under review.

computed tractably. Additionally, score-based models can be optimized with the score function directly through score-matching [3], unlike variational autoencoders which are optimized with Evidence-Based Lower Bound (ELBO) for the data distribution. Lastly, training score-based models do not involve min-max optimization like GANs; as a result, training score-based models are more stable and do not suffer from mode collapse. Because of these reasons, diffusion and score-based models have gained popularity in the machine learning community recently.

In this project, we explore the capability of recently introduced score-based generative models ready pre-trained to solve unconditional image generation. We combine these models with Open's AI CLIP which evaluates the probabilities of image-text pairs based on the semantic meanings, i.e. if text well describes the image, CLIP will predict high probability. With CLIP, we can control the sampling processes to generative images based on input text prompts. To tackle text-conditioned image synthesis, we explore two approaches to combine diffusion models and CLIP: (1) Contrastive conditional diffusion sampling, and (2) CLIP-guided unconditional diffusion sampling. Our code is available at https://github.com/patinkaew/sde.txt.

2 Related Work

Text-guided image synthesis requires a multimodal model that encapsulates the joint probability of both images and texts. For example, [4] uses deep Boltzman machine to jointly model image and text tags. With the introduction of generative adversarial network (GAN) [5] that can generate photorealistic images very well, many researchers have applied GAN to the text-guided image synthesis task. The first in this line of work is [6], which conditions a deep convolutional GAN on text features to synthesize images. Then, [7] introduced the attention mechanism to the GAN model, improving the precision of the image generation in capturing fine-grained semantic content in the texts.

While GAN-based methods can produce high-quality images, there are some drawbacks, such as training difficulty due to mode collapse. Recently, Song et al. introduced score-based generative model [3], which can be trained more easily than GAN and offer competitive performance. The score-based approach models the gradient of the log-likelihood, i.e. *score*, and sample from the model using Langevin dynamics [8, 9]. Score-based modeling also offers a natural way to perform conditional generation, so it is suitable for the text-guided image synthesis task. While some works have used score-based models to conditionally generate images from classes [10, 2], generation using score-based models by conditioning on texts has not been extensively studied yet.

In addition to conditional generation, there exist alternative approaches to control the generative process of score-based models. One such work most related to ours is DiffusionCLIP [11], in which CLIP is used to calculate the loss between the generated image and the text caption, and then the loss is backpropagated to update the model parameters or the inputs to nudge the image generation. We take a similar approach in this work.

3 Backgrounds

Our approach combines denoising diffusion probabilistic model (DDPM) [12] with CLIP [13] to control image generation using text captions. In this section, we give a concise overview of DDPM and explain how to sample from DDPM deterministically via DDIM.

3.1 Denoising Diffusion Probabilistic Model (DDPM)

The concepts and notations from this subsection are based on [14, 11]. DDPMs are a kind of a latent variable model where data points are mapped onto a latent space via a Markov chain. In the forwrd process, data points with some distribution $q(\mathbf{x})$ is mapped to a latent space following prior distribution of choice, e.g. a Gaussian distribution. When the forward process is a Markov chain with Gaussian transitions, as in [12], we may express each forward step as follow:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I});\ t = 1, \dots, T$$

for some choice of an increasing sequence of variance schedule $\{\beta_t\}_{t=1}^T$. We denote $\bar{\alpha}_t = \prod_{i=1}^t (1-\beta_i)$. For this forward process, it follows that $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})$, or, equivalently,

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\mathbf{w},\ \mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}).$$

Observe that $\mathbf{x}_T$ approaches $\mathcal{N}(\mathbf{0}, \mathbf{I})$ as $t \to T$ since α_T is typically close to 0 for any $\mathbf{x}_0$, as expected. Then, we learn the reverse process which maps the latent space to the data distribution $q(\mathbf{x})$ by maximizing a variational lower bound of the log likelihood $p_\theta(\mathbf{x})$ approximating $q(\mathbf{x})$. For the parameterization of the reverse process $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \sigma_\theta^2(\mathbf{x}_t, t)\mathbf{I})$, the training objective can be simplified as

$$\min_\theta \mathcal{L} = \min_\theta \mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}), \mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})} \left\| \mathbf{w} - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right\|_2^2.$$

Once trained, the model $\boldsymbol{\epsilon}_\theta$ can be used to sample data points from the latent via a stochastic reverse process:

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\bar{\alpha}_t}} \left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z};\ t = T, \dots, 1. \tag{1}$$

Next, we relate DDPMs to score-based models. [2] has shown that DDPMs are equivalent to score-based models with variance-preserving (VP) noise schedule. Note that a score-based model $s_\theta(\mathbf{x})$ approximates the Stein's score function $\nabla_\mathbf{x} \log p(\mathbf{x})$. The VP score-based model is related to DDPM as follow:

$$s_\theta(\mathbf{x}_t, t) \approx \nabla_{\mathbf{x}_t} \log p_\theta(\mathbf{x}_t) = -\frac{1}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(\mathbf{x}_t, t). \tag{2}$$

The relationship between DDPMs and score-based models is key to one of our methods which relies on conditional generation.

3.2 DDIM: Sampling DDPM Deterministically

A different way to sample from $\boldsymbol{\epsilon}_\theta$ is by using the Denoising Diffusion Implicit Model (DDIM) approach [14]. DDIM uses a different forward process from DDPM that has the same forward marginals:

$$\mathbf{x}_{t+1} = \frac{\sqrt{\bar{\alpha}_{t+1}}}{\sqrt{\bar{\alpha}_t}} \left(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t} \epsilon_\theta(\mathbf{x}_t, t) \right) + \sqrt{1-\bar{\alpha}_{t+1}} \epsilon_\theta(\mathbf{x}_t, t);\ t = 0, \dots, T-1.$$

For sampling, the reverse process can be calculated using the same model $\boldsymbol{\epsilon}_\theta$ as follow:

$$\mathbf{x}_{t-1} = \frac{\sqrt{\bar{\alpha}_{t-1}}}{\sqrt{\bar{\alpha}_t}} \left(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t} \epsilon_\theta(\mathbf{x}_t, t) \right) + \sqrt{1-\bar{\alpha}_{t-1}} \epsilon_\theta(\mathbf{x}_t, t);\ t = T, \dots, 1. \tag{3}$$

DDIM is deterministic, so it is crucial in works like DiffusionCLIP [11] in which the gradient is backpropagated through the reverse process, because the stochastic reverse process (1) is noisy and hard to be optimized. DDIM is used in one of our methods which has backpropagation through the reverse process.

4 Methods

We propose two methods for text-guided image synthesis. The first method is based on conditional sampling using score-based generative models. The second method uses a CLIP loss to finetune the input to the sampling process, $\mathbf{x}_T$, in order to guide the generation.

4.1 Contrastive Conditional Sampling

Score-based generative modeling naturally allows for conditional sampling by changing unconditional score function $\nabla_{\mathbf{x}} p_{\text{data}}(\mathbf{x})$ to conditional score function $\nabla_{\mathbf{x}} p_{\text{data}}(\mathbf{x}|\mathbf{y})$. To sample from $\nabla_{\mathbf{x}} p_{\text{data}}(\mathbf{x}|\mathbf{y})$, we factor it into the gradient of the prior probability and the gradient of the conditional probability:

$$\nabla_{\mathbf{x}} p_{\text{data}}(\mathbf{x}|\mathbf{y}) = \nabla_{\mathbf{x}} p_{\text{data}}(\mathbf{x}) + \nabla_{\mathbf{x}} p_{\text{data}}(\mathbf{y}|\mathbf{x}). \tag{4}$$

In our case, $\mathbf{x}$ is the image, and $\mathbf{y}$ is the text. The first term can be modelled by unconditioned generative models, and the second term can be estimated with text-captioning models.

To estimate the conditional term $\nabla_{\mathbf{x}} p_{\text{data}}(\mathbf{y}|\mathbf{x})$, we use probabilities computed by CLIP model [13]. CLIP (Contrastive Language Image Pre-training) is a model optimized through contrastive learning consisting of a text encoder and an image encoder. These CLIP encoders will produce close feature representations if text and image are paired by semantic meanings. That is, given an image and a group of text prompts, CLIP can indicate which text prompt best matches with the image. The contrastive behavior of CLIP motivates the usages of multiple text prompts with one target and multiple distractors.

Ideally, the domain of the input caption $\mathbf{y}$ is all possible text captions. However, to make computation tractable, we choose only a few words when calculating the conditional probability. For example, given an image $\mathbf{x}_t$ and a list of texts "airplane", "dog", and "cat" where only "airplane" is our target caption, we calculate the normalized probability of each text caption over all three captions given the image using CLIP. We then use the normalized conditional probability $p_{\text{CLIP}}(\mathbf{y}|\mathbf{x}_t) = p_{\text{CLIP}}(\text{airplane}|\mathbf{x}_t)$ in our sampling process. As a result, CLIP will guide the image generating procedures to the target prompt. Throughout the sampling process, the probability of the target caption will increase while the probabilities of the distractor captions will decrease. Using the relationship between score-based generative modeling and DDPM in equation (2), we modify the DDPM reverse process (1) to obtain a new update

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathbf{x}_t, t) + \gamma\beta_t \nabla_{\mathbf{x}_t} \log p_{\text{CLIP}}(\mathbf{y}|\mathbf{x}_t)\right) + \sigma_t \mathbf{z};\ t = T, \ldots, 1, \tag{5}$$

where $p_{\text{CLIP}}(\mathbf{y}|\mathbf{x}_t)$ is the conditional probability $p_{\text{data}}(\mathbf{y}|\mathbf{x}_t)$ approximated using CLIP., and γ a scaling hyperparameter used to adjust the balance between uncondtional and conditional terms.

4.2 CLIP-guided Unconditional Sampling

Assuming that we have a trained model $\boldsymbol{\epsilon}_\theta$, we can generate images with text guidance by Algorithm 1 as summarized in Figure 1. We sample an image unconditionally from a latent $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ using $\boldsymbol{\epsilon}_\theta$ according to the DDIM reverse process 3. Then, we calculate a CLIP loss between the generated image and the target text, and backpropagate the loss to update the latent $\mathbf{x}_T$. We choose the DDIM

Figure 1: The CLIP-guided image sampling procedure.

reverse process (3) over the DDPM reverse process (1) because the randomness in the DDPM reverse process can make latent optimization difficult [11].

CLIP [13] comprises of an image encoder and a text encoder which map to the same latent space, and they are trained such that the image latent vector will be similar to the text latent vector if they share an underlying semantic meaning. Following [15], we use the cosine distance between the image embedding and the text embedding from CLIP as the loss:

$$\mathcal{L}_{CLIP}(\mathbf{x}, \mathbf{t}) = 1 - \frac{\langle I(\mathbf{x}), T(\mathbf{t}) \rangle}{\|I(\mathbf{x})\|_2 \, \|T(\mathbf{t})\|_2}, \tag{6}$$

where I is the CLIP image encoder, T is the CLIP text encoder, $\mathbf{x}$ is an image, and t is text input.

Note that this approach requires backpropagating through all steps of the reverse process, and a typical number of steps is $T = 1000$. However, backpropagating through 1000 steps entails a very large memory footprint. To make computation tractable, we have to reduce the number of steps to $S_{gen} \ll T$ while keeping the same noise variance schedule β_t as a function of time t. In other words, in practice, we replace T in Algorithm 1 with S_{gen}.

Algorithm 1: CLIP-guided image generation from DDPM

Input: Embedding of the target text $\mathbf{t}$, DDPM $\boldsymbol{\epsilon}_\theta$, step size η.
Output: A sampled image $\mathbf{x}_T$.
$\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
while $\mathbf{x}_T$ *has not converged* **do**
  Calculate the image $\mathbf{x}_0$ from the DDIM reverse process (3) starting from $\mathbf{x}_T$ using model $\boldsymbol{\epsilon}_\theta$.
  Calculate $\mathcal{L}_{CLIP}(\mathbf{x}_0, \mathbf{t})$ as in equation (6).
  Calculate the gradient $\partial \mathcal{L}_{CLIP}(\mathbf{x}_0, \mathbf{t}) / \partial \mathbf{x}_T$ by backpropagation.
  Take an optimizer step (with SGD, Adam, etc.) to update $\mathbf{x}_T$ in order to minimize the loss $\mathcal{L}_{CLIP}(\mathbf{x}_0, \mathbf{t})$.
end
return $\mathbf{x}_T$

5 Experiments and Results

We developed our code and ran all experiments on Google Cloud Platform with N1 machine and a single Nvidia Tesla K80 GPU.

5.1 Contrastive Conditional Sampling

For Contrastive Conditional sampling, we used the DDPM trained on the CIFAR-10 dataset [16] from [17], which is a PyTorch reimplementation of the original DDPM work [12]. We use the same model configurations from DDIM repository [14]. In particular, the noise variance schedule β_t linearly increases from 0.0001 to 0.02, and number of diffusion time steps is $T = 1000$.

We benchmark the performance of contrastive conditional sampling method with unconditional sampling and spherical distance sampling introduced in the Colab notebook *Quick CLIP Guided Diffusion*. The spherical distance loss is given by

$$\mathcal{L}_{\text{spherical}}(\mathbf{x}, \sqcup) = 2 \arcsin \left[\frac{1}{2} \left(\frac{I(\mathbf{x})}{||I(\mathbf{x})||_2} - \frac{T(\mathbf{t})}{||T(\mathbf{t})||_2} \right) \right]^2$$

where I is the CLIP image encoder, T is the CLIP text encoder, $\mathbf{x}$ is an image, and t is text input. We use this loss to model conditional score function, $\nabla_{\mathbf{x}} p(\mathbf{y}|\mathbf{x}) = -\nabla_{\mathbf{x}} \mathcal{L}_{\text{spherical}}(\mathbf{x}, \sqcup)$ similar to the formulation in referred Colab notebook. Notice that conditional sampling through spherical distance only utilizes the single target prompts while contrastive conditional sampling uses both target text and distractors.

For all three sampling methods, we generate 100 samples. To control the performance across all methods, we start with the same noisy images for all three sampling procedures. We use “airplane"

as target text and the remaining nine class labels of CIFAR 10 dataset as the distractors. We use sampling time and Kernel Inception Distance (KID) as the performance metrics. Lower sampling time indicates better performance. Since KID uses Maximum Mean Discrepancy (MMD) which is a two-sample test statistics that measures the difference of two distributions, if we use real images from data distribution and generated images from our diffusion model, lower KID indicates more realistic images with respect to the given real image reference. After hyperparameter search, we found that $\gamma \approx 100$ works best for contrastive sampling, and $\gamma \approx 1000$ works best for spherical distance sampling. During sampling processes, we also monitor CLIP probabilities for contrastive and spherical distance sampling.

Figure 2: Sample images generated from different methods: unconditional (row 1), spherical distance (row 2), and contrastive (row 3)

Figure 2 shows generated samples from different sampling procedures. The sample's quality is not very high due to the pre-trained generative model as seen in unconditional sampling. We can see that both contrastive and spherical distance can perturb noisy image to given target text input "airplane". In some cases, we observe incorrect image generation for both conditional sampling methods. After monitoring CLIP probabilities, we observe that CLIP classify these final images as "airplane" even though they are not correct visually. This indicates the limitation on the usage of CLIP model to estimate the conditional probability during diffusion processes.

Sampling method	Contrastive conditional sampling	Spherical distance conditional sampling	Unconditional sampling
Time to generate 100 samples (minutes)	90	117	20
KID	0.0618 ± 0.0210	0.0653 ± 0.0236	0.1184 ± 0.0333

Table 1: Performance metrics calculated for all three sampling procedures

We report the two performance metrics in Table 1. In particular, we use real images from the "airplane" class in CIFAR 10 as a reference for evaluating KID. We can see that both conditional methods achieve a better KID score than unconditional sampling as expected. We see that contrastive sampling also achieves slightly better performance compared to spherical distance sampling. Additionally, we see substantially lower generation time for contrastive sampling than spherical distance sampling.

An example of CLIP probabilities for contrastive and spherical distance sampling during sampling processes are shown in figure 3. We can see that the probabilities of each class start similar for both conditional sampling methods since we start with the same noisy image for both methods. However, we can see that the probability of the target prompt increases much more rapidly for contrastive sampling compared to spherical sampling. From the results, our experiment shows potential for better performance of contrastive sampling over the spherical distance baseline method.

5.2 CLIP-guided Unconditional Sampling

For CLIP-guided unconditional sampling, we used the DDPM trained on the CIFAR-10 dataset [16] from [17], which is a PyTorch reimplementation of the original DDPM work [12]. The noise variance

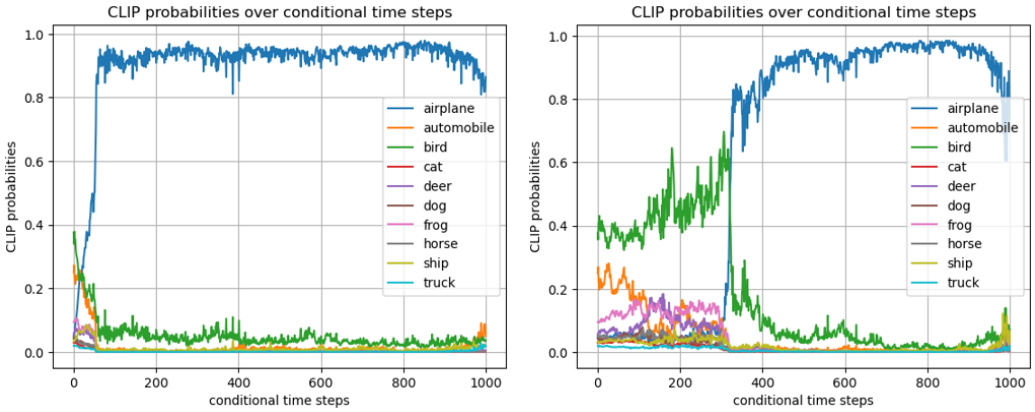

Figure 3: Probabilities for each class computed by CLIP model for contrastive (left) and spherical distance (right)

schedule β_t linearly increases from 0.0001 to 0.02. The default number of time steps for this model is $T = 1000$, but we chose $S_{gen} = 50$ so that our GPU could accommodate the memory footprint from the gradient calculation. We used "car" as the text prompt. Following [11], we used the Adam optimizer [18] with the learning rate 10^{-6} to optimize the input latent $\mathbf{x}_{S_{gen}}$.

We observed that the CLIP-guided unconditional sampling approach struggles to generate meaningful images. As shown in Figure 4, the generated images appear to be mostly noise. This result can be accounted by many factors. First, using too small S_{gen} may reduce the quality of the generated image such that the DDIM sampling process (3) produces noise-like images. Second, unlike the image editing method [11] in which an image first undergoes the forward process to obtain a latent, our initial latent $\mathbf{x}_T$ is a Gaussian noise with no corresponding image. Even though starting from some Gaussian random $\mathbf{x}_T$ can produce high-quality images when $T = 1000$, we can take only S_{gen} steps, so it is much more difficult to generate good images. Third, CLIP was not trained to model the probability of images with high noise. Hence, if the generated image $\mathbf{x}_0$ in the first iteration of latent optimization is very noisy, CLIP will not yield a meaningful loss. A combination of these three factors make the CLIP-guided unconditional sampling approach very challenging.

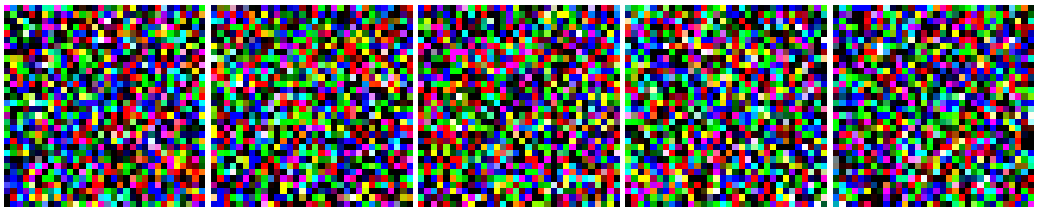

Figure 4: Images generated using the CLIP-guided unconditional sampling approach. The number of steps S_{gen} is 50, and the text prompt for all images is "car".

6 Conclusion

Our project demonstrates the capability of combining publicly available pre-trained generative models to tackle text-guided image synthesis tasks. Specifically, due to their advantages and current popularity in the machine learning research community, we use recently-introduced diffusion/score-based models such as DDPM [12] and DDIM [14] pre-trained to generate images unconditionally. To control the generation processes to condition on the given prompt, we use CLIP model [13] to estimate the conditional probability between generating images and given text prompts.

We explore two approaches in this project. The first approach "contrastive conditional sampling", motivated by the contrastive nature of CLIP, uses a target prompt together with multiple distractor prompts to model the conditional probability during diffusion processes. The contrastive conditional

sampling shows a better performance - lower sampling time and lower KID - compared to the baseline spherical distance sampling method.

The second approach “CLIP-guided unconditional sampling" indicates the insight to the success of DiffusionCLIP [11] model: starting with the real images is crucial. Since DiffusionCLIP tackles the image-editing problem, DiffusionCLIP has access to the real images. However, in our case, there is no obvious way to generate starting noisy images given the input prompt. We show that starting with completely random noise, our model struggle to generate meaningful images. Due to hardware limitations, we cannot run diffusion processes for many steps, so the intermediate images fed to the CLIP model are mostly noise. CLIP probabilities are not reliable in this case since CLIP was not trained to model the probability of images with high noise.

For future works, for the “contrastive conditional sampling", we are interested in conducting experiments on more complex text prompts, not simply the class labels, for example “red truck". We can also experiment on diffusion models trained on higher resolution dataset such as ImageNet [19] or CelebA-HQ [20]. For the “CLIP-guided unconditional sampling", we are interested in experimenting on knowledge distillation models which reduce the number of diffusion steps substantially [21]. With this model, we do not require a large number of steps to generate meaningful images, so CLIP-guided unconditional sampling is more applicable. Finally, we can combine these two approaches. In our preliminary experiments on spherical distance conditional sampling, we observe that the quality of the final images strongly depends on the starting noisy images. As a result, if we can start with better and more structured noisy images, we can archive better performance. We can generate noisy images from CLIP-guided unconditional sampling. Then, we diffuse these generated noisy images using contrastive conditional sampling to obtain the final images.

7 Acknowledgement

This project would not be possible without supports from many individuals. We would like to give special thank to our TA project mentor Kelly He for her guideline throughout the project development. We would like to thank all CS 236 instructors and staff for this opportunity, thoughtful instructions, and great supports.

References

[1] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. *CoRR*, abs/2102.12092, 2021.

[2] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations, 2021.

[3] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution, 2020.

[4] Nitish Srivastava and Russ R Salakhutdinov. Multimodal learning with deep boltzmann machines. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[5] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[6] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis, 2016.

[7] Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. Attngan: Fine-grained text to image generation with attentional generative adversarial networks, 2017.

[8] G. Parisi. Correlation functions and computer simulations. *Nuclear Physics B*, 180(3):378–384, 1981.

[9] Ulf Grenander and Michael I. Miller. Representations of knowledge in complex systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, 56(4):549–603, 1994.

[10] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis, 2021.

[11] Gwanghyun Kim and Jong Chul Ye. Diffusionclip: Text-guided image manipulation using diffusion models, 2021.

[12] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020.

[13] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020, 2021.

[14] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models, 2021.

[15] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. Styleclip: Text-driven manipulation of stylegan imagery, 2021.

[16] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research).

[17] Patrick Esser. Pytorch pretrained diffusion models, 2020. GitHub Repository, https://github.com/pesser/pytorch_diffusion. Accessed 30 Nov 2021.

[18] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[19] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[20] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[21] Eric Luhman and Troy Luhman. Knowledge distillation in iterative generative models for improved sampling speed, 2021.