Stable Walk: An interactive environment for exploring Stable Diffusion outputs

Mattias Rost [2] and Sebastian Andreasson[1]

[1] *University of Gothenburg, Gothenburg, Sweden*

Abstract

The past year saw an advancement in text-to-image models. Several models were released as well as services made available for users to use to generate images. These have become popular because without special training, the models can generate images from a simple text prompt. However the parameter space of these models go beyond the text prompt, and skilled users can finetune the output of the models using these parameters. In this work we present ongoing work developing a tool to explore the parameter space of Stable Diffusion. The aim of the tool is to make it possible to explore the parameter space visually. In particular we present a novel way of exploring the text embedding space by allowing users to combine several prompts.

Keywords

ui, interactive tool, stable diffusion

1. Introduction

The advances in text to image models has rendered an explosion in creative uses of such models. In 2022, Google announced Imagen [1] and OpenAI announced Dall-E 2 [2]. A few months later Midjourney announced an open beta of their model (https://midjourney.com/), and StabilityAI announced Stable Diffusion [3]. While Imagen is not publicly available, Dall-E 2 is available through an API for a small cost and Midjourney is available to use through Discord for a monthly fee (although using a freemium model allows use for free to some extent). Stable Diffusion on the other hand is freely available to run locally on your own machine. This has resulted in a number of online versions of Stable Diffusion with varying pricing models and features (e.g. DreamStudio).

The power of these models come from the fact that they are able to generate realistic images from a single text prompt. Entering the typical example prompt "an astronaut riding on a horse" will generate an endless stream of variations of astronauts riding on horses. Crafting prompts that render good looking images have become an art in itself, such that it is now possible to sell and buy them (https://promptbase.com/).

But it is not only possible to steer the output through text, but the algorithms used also allow for more parameterisation. Initial seed, guidance scale, and inference steps, are all different parameters to the algorithm that affect the output in different ways. However, it is not as immediately clear how these and other parameters can affect the output for new users.

In this position paper we present a web based tool that allows users to explore the parameter space of Stable Diffusion. In particular, it opens up Stable Diffusion to explore the *text embedding* space. The text embeddings are the vector representations of the prompts that are the actual input to the diffusion algorithm to condition its output. The tool lets users visually map a small part of the text embedding space, and to generate images from within this space. The aim of the tool is to allow exploration of the different inputs and configurations of Stable Diffusion in an active and

Joint Proceedings of the ACM IUI Workshops 2023, March 2023, Sydney, Australia
EMAIL: mattias.rost@ait.gu.se (A. 1); sebastian.andreasson@ait.gu.se (A. 2)
ORCID: XXXX-XXXX-XXXX-XXXX (A. 1); XXXX-XXXX-XXXX-XXXX (A. 2); XXXX-XXXX-XXXX-XXXX (A. 3)

engaging way. Other tools allow the user to set parameters, but our tool makes this exploration simpler.

2. Related Work

Text to image models are not the first types of algorithm used to create content in a seemingly automatic fashion. For instance procedural content generators have been around in computer graphics for a long time. Procedural content generators typically involve mathematical formulas to generate e.g. trees and landscapes, using fractals or Perlin noise. These techniques are commonly used in game production.

It has been argued that using such techniques is not widely understood, but rather seen as magic. In order to combat this, researchers developed Danesh [4]. It allows users of Unity to explore the procedural generators distribution space as well as "automatically searching the parameter space for configurations that produce a specific outcome" (ibid). It simplifies the ability for humans to co-create with the procedural generator.

The use of deep learning techniques is however more recent. GANs (generative adversarial networks) have dominated as the main technique for image generation. Developed in 2016, they have been shown to be able to generate realistic images from a training dataset, such as faces [5]. At last years workshop on HAI-GEN, Grabe et al. presented a framework for co-creativity using GANs [6]. They showed four interaction patterns that applications implementing GANs allow when used in co-creation between humans and the GAN.

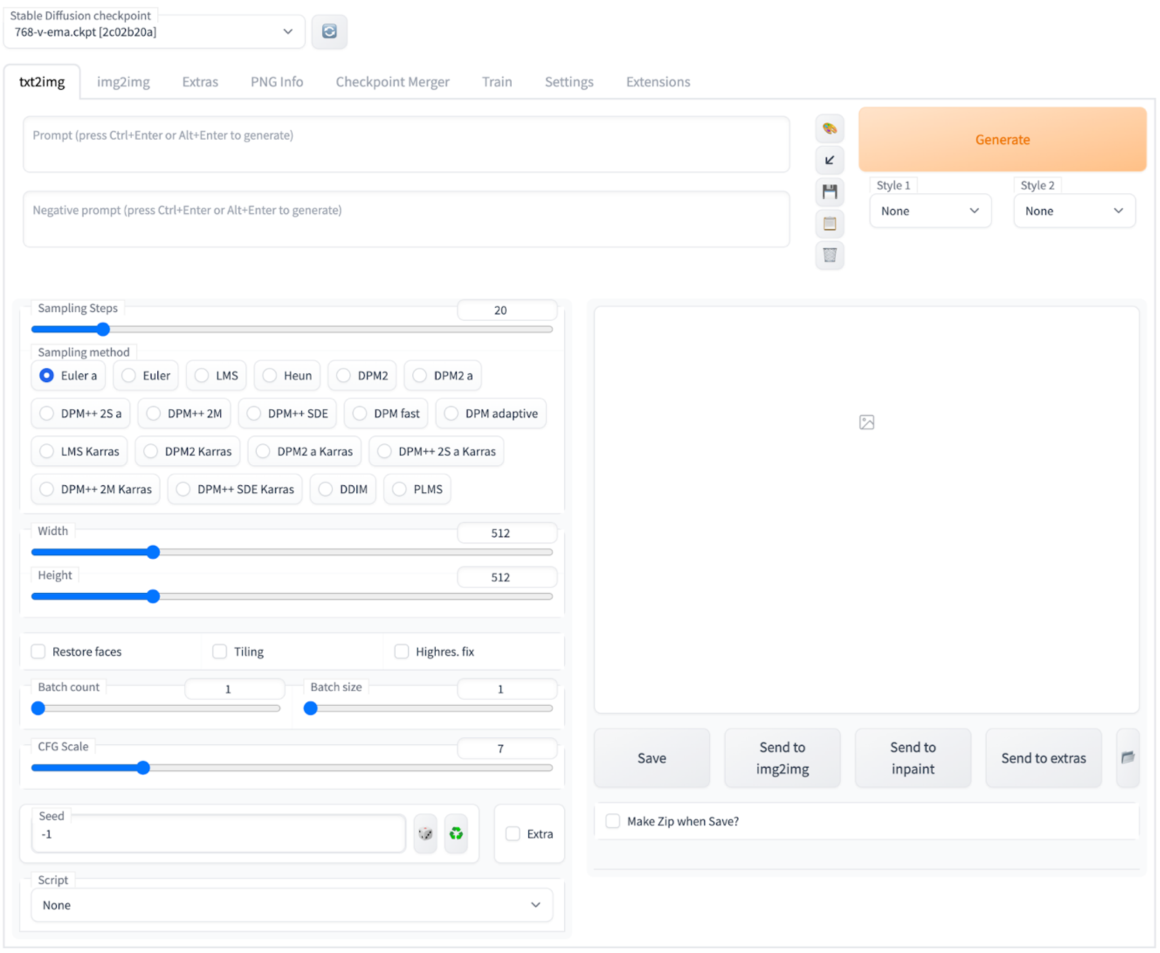
Today's diffusion text-to-image models differ from the typical GAN in important ways. GANs are trained on datasets and then sampled from to generate images like the ones described from the training dataset. While there are conditional GANs and they are in other ways parameterised to allow some control, they do not allow the same kind of control as e.g. SD. Instead SD is pre-trained on a wide range of images from which images can be sampled by describing the image in text (conditioning). This would mean the use of these models in co-creation differ in important ways from using GANs as described by Grabe et al.. From the four interaction patterns, we would argue that the typical use of SD involves no curation, but instead a lot of exploring. The way we understand their pattern of conditioning, it refers to a more strict form of conditioning than how SD is conditioned. The use of diffusion models are better described as a combination of exploring and conditioning. Our tool is one way of opening up the means for doing this exploring and conditioning.

2.1. Stable Diffusion UIs

There are now several online services available that let users generate images using different versions of SD. DreamStudio (dreamstudio.ai), Stable Diffusion Online (stablediffusionweb.com), as well as Hugging Face (https://huggingface.co/spaces/stabilityai/stable-diffusion) offer ways for users to try SD for free. They all take a text prompt, and render images. They let you customize other parameters as well such as *cfg scale*, and *steps* (explained later). These services make this model available to anyone to use.

It is also possible to run SD locally by downloading the model, and Hugging face has integrated it into their libraries for ease of use. Developers have also implemented UIs on top of these libraries, where the most popular one is Stable Diffusion Web UI (*WebUI*) (https://github.com/AUTOMATIC1111/stable-diffusion-webui). WebUI aims to make all possible parameters available to users into an all encompassing UI. As can seen in Figure 1, the UI is composed of text boxes, sliders, and radiobuttons. While this makes the options available, such controls do not make them intuitive. This is not necessarily negative, since WebUI does not aim to create an intuitive interface for using Stable Diffusion. However, to explore e.g. the effect of a parameter, the user has to manually change the parameter and generate a new set of images with that setting.

While it is possible to create a X/Y plot varying two parameter values, it forces the user to generate a large array of images in one go, rendering the exploration process less active, where the process of analysing the change becomes retroactive rather than proactive.



Stable Diffusion checkpoint: 768-v-ema.ckpt [2c02b20a]

Style 1: None

Style 2: None

Sampling Steps: 20

Sampling method: [✓ Euler a]

Width: 512

Height: 512

Batch count: 1

Batch size: 1

CFG Scale: 7

Seed: -1

Script: None

Figure 1. A screenshot of the WebUI main interface, illustrating the layout and components of the UI.

2.2. Inner Works of Stable Diffusion

Next we will describe some of the inner works of Stable Diffusion. The purpose here is not to describe it in detail, but to make some of the components visible such that they can be referred to in later sections.

Stable diffusion (SD) is a latent text-to-image diffusion model [3]. Essentially a model is trained to predict noise in an image, such that we can remove that noise from the noisy image. The result is that an image can be generated from pure noise, by gradually removing more and more noise such that an image appear from the noise. The model is conditioned and guided with a text prompt such that the resulting image can be described by the text prompt.

In order to accomplish this, SD consists of three components: A variational autoencoder (VAE), a noise prediction model, and a text encoder. The VAE is used to convert a high dimension image into a lower dimensional latent space, and from the lower dimension latent space into image space. The text encoder is used to turn a text prompt into a text embedding. As a text encoder, version one of SD uses CLIP from OpenAI [7], whereas version two uses OpenCLIP [8]. These encoders are trained such that the embedding of a text description is close to the embedding of an image described by the description.

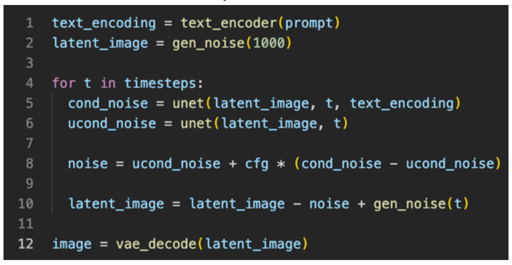
During training SD is trained on 1000 diffusion steps but during inference you can choose how many steps you want to use to denoise the image. More steps generally creates a higher-quality image but takes longer than using fewer steps. Users must choose steps to balance speed and quality.

The whole process can be described by the pseudo code in figure 2. First the text encoding and initial noise is created. Then through a number of timesteps we ask the model to predict both conditioned and unconditioned noise in the latent image. Line 8 implements classifier free guidance [9]. Line 10 removes the predicted noise from the latent image, and adds noise for the current timestep. At the last timestep it adds no noise. Finally the latent image is decoded by the VAE to create the final image.

The exact implementation of *gen_noise*, *line 10*, and *timesteps* depends on an algorithm for how to do the reverse diffusion process, e.g. DDPM[10], DDIM[11], and LMS [12]. In this work we use LMS.

While the text encoder, Unet, and VAE are pretrained and given by SD, there are certain parameters available for developers and users to set. This paper presents a novel UI for exploring this parameter space in terms of output from SD.

From the code, it should be clear that changing the timesteps, cfg, and text_encoding should have an impact, but it is not clear what the impact will be. Our tool help with that understanding through a visual user interface, that we call *Stable Walk*.



```
1   text_encoding = text_encoder(prompt)
2   latent_image = gen_noise(1000)
3
4   for t in timesteps:
5     cond_noise = unet(latent_image, t, text_encoding)
6     ucond_noise = unet(latent_image, t)
7
8     noise = ucond_noise + cfg * (cond_noise - ucond_noise)
9
10    latent_image = latent_image - noise + gen_noise(t)
11
12  image = vae_decode(latent_image)
```

Figure 2. Pseudo-code listing of generating images using Stable Diffusion.

3. Stable walk

In order to generate images using SD, the most straightforward interface is the text prompt. Users can type in a line of text and the output is an image. However, in order to get better control of the output, the diffusion process takes a number of parameters that users can change. E.g. we can change the number of diffusion steps taken. Picking 100 steps instead of 10 typically generates higher quality images, but takes 10 times as long to produce. Changing the classifier free guidance scale changes other aspects of the image. In more advanced tools, these parameters are available as simple inputs. In order to get a better understanding of the impact of these parameters, users must explore the output while varying the parameters.

We therefore created a tool that let users more visually explore the parameter space through the outputs of SD. The tool is web based and interacts with SD through a custom API. In our setup the API is run on a server with a RTX 3090 graphics card that renders images on average just over a second.

The web interface has two tabs, and under each tab you can vary parameters in different ways. These are: Grid and Canvas. Both tabs have some controls in common. The controls in common are: a base prompt that is added to all prompts; a negative prompt used for negative conditioning; and a seed for the random number generator.

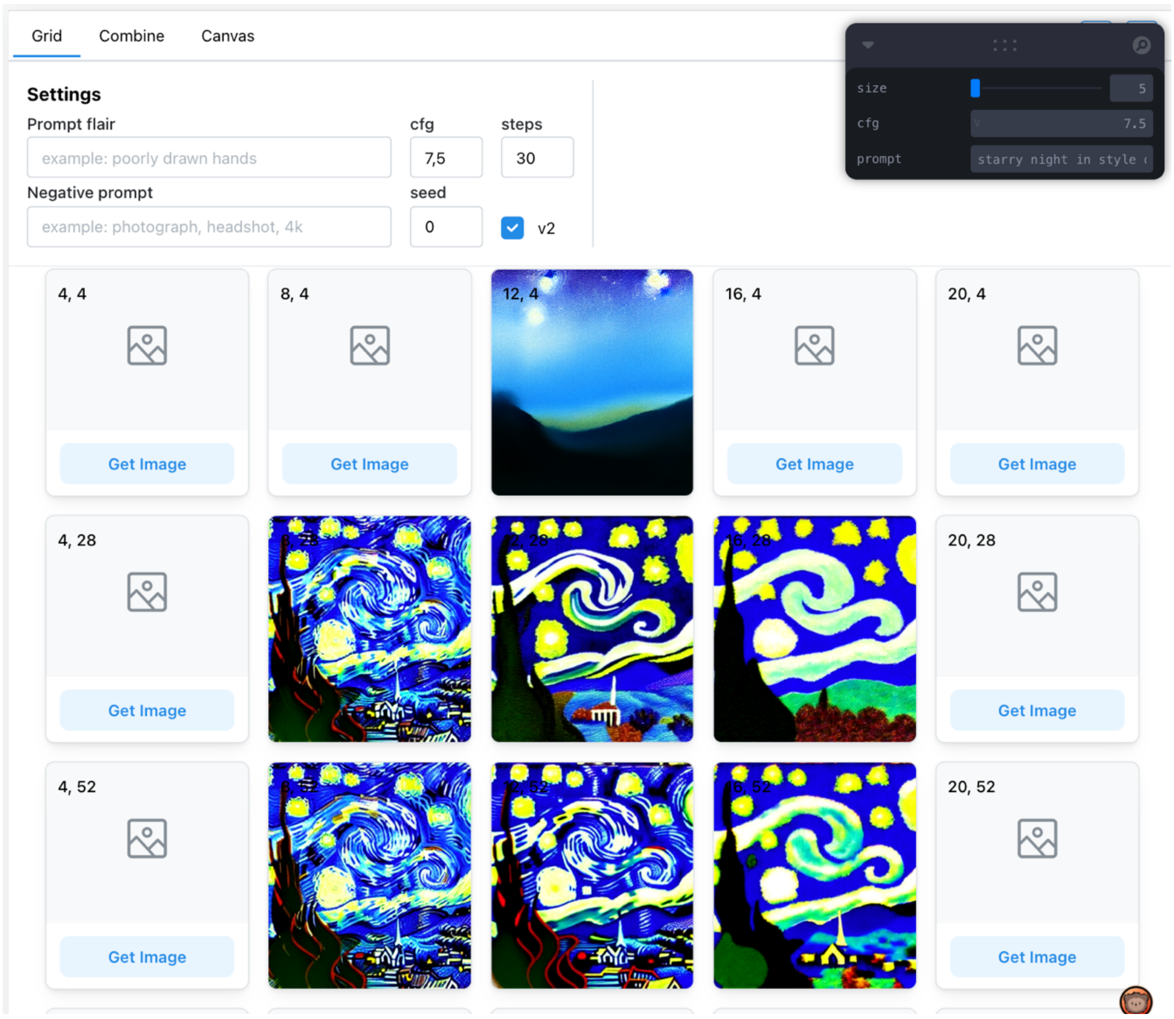
3.1. Grid

In the grid view the user can choose to generate images in a grid from a prompt. Positions in the grid determine the values of *cfg* and *steps*. The user can choose the number of grid positions and thus the difference in value between grid positions. The row in the grid determines *step*, and the column determines *cfg*. Top row is set to 4 steps and the bottom row is set to 100 steps. The left most column sets cfg to 4 and the right to 20. These values were chosen empirically to span a variety of outputs.

Initially the grid is showing placeholder images. By clicking on an image, a request is sent to the server that generates the image with the given parameters. To explore how changes in cfg and/or steps affect the output, the user selects images in the grid by either taking small or big steps in the grid. This enables the users to actively traverse the parameter space to explore what images are generated by the algorithm, as shown in figure 3.

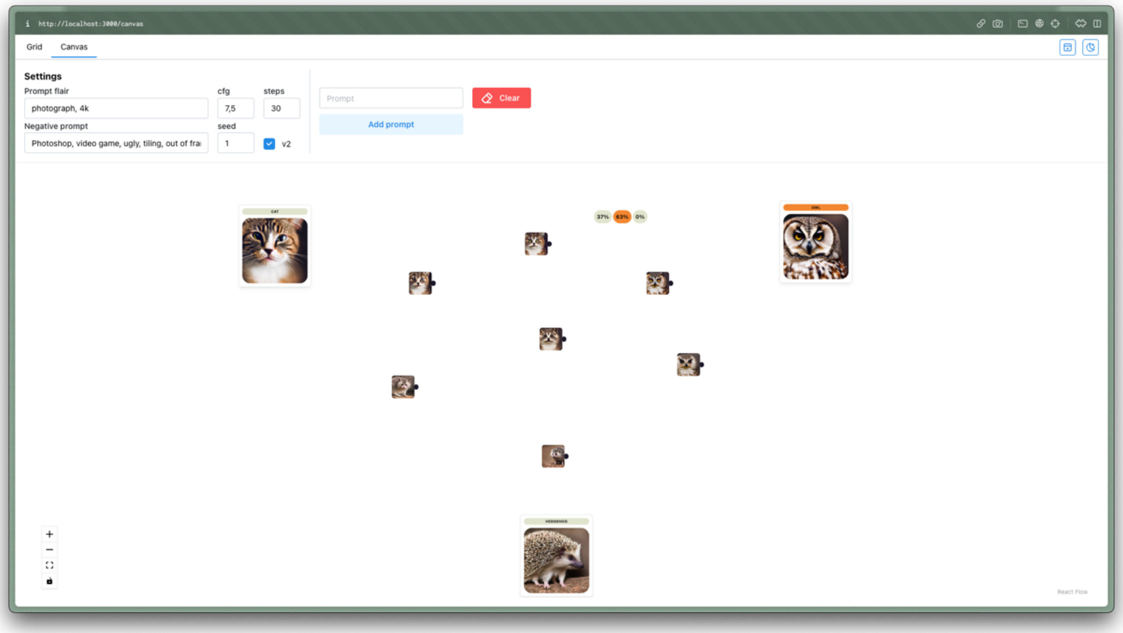
3.2. Canvas - exploring text embedding space

In the Canvas tab we are exploring another parameter space, namely that of text embeddings. The UI consists of an infinite pan and zoomable canvas. The user starts by adding prompts that can be placed onto the canvas freely. Once placed on the canvas an image is generated and shown on the canvas. Once a few prompts have been placed on the canvas, the user can then generate images that are combinations of these prompts, by clicking on a point between the prompts.



Prompt flair: example: poorly drawn hands
cfg: 7,5
steps: 30
Negative prompt: example: photograph, headshot, 4k
seed: 0
✓ v2

Figure 3. Grid view with prompt “Starry night in style of monet”.



Prompt flair: photograph, 4k
cfg: 7,5
steps: 30
Negative prompt: Photoshop, video game, ugly, tiling, out of fra
seed: 1
✓ v2

Figure 4. The UI of Canvas, showing three prompts “Cat”, “Owl”, and “Hedgehog” laid out in three corners of a triangle. Seven images have been generated as combinations of those three prompts.

The combinations are generated by calculating a new text embedding from the text embeddings of the existing text prompts. This is calculated by taking a linear combination of the embeddings, where the weights correspond to the relative position of the point on the canvas in relation to the prompts. More formally, if the points of the prompts on the canvas are the 2d coordinates p_i, and the target point (2d coordinate of the mouse cursor) is t. Then we find the scalar weights w_i such that:

t = sum(w_i * p_i), and sum(w_i) = 1.

The target text embedding, *e*, is then calculated from the text embeddings of each prompt, pe_i, according to

e = sum(w_i * pe_i)

By moving the target point closer to a particular prompt, the intuition is that the resulting embedding will contain more of that prompt and less of the others.

By clicking on the canvas, new images are generated in the position of the canvas corresponding to the weights of that position. By clicking on the canvas, the user essentially samples the embedding space between the entered prompts. By zooming in on the canvas, it makes it easy to make very fine tuned transitions between weight values.

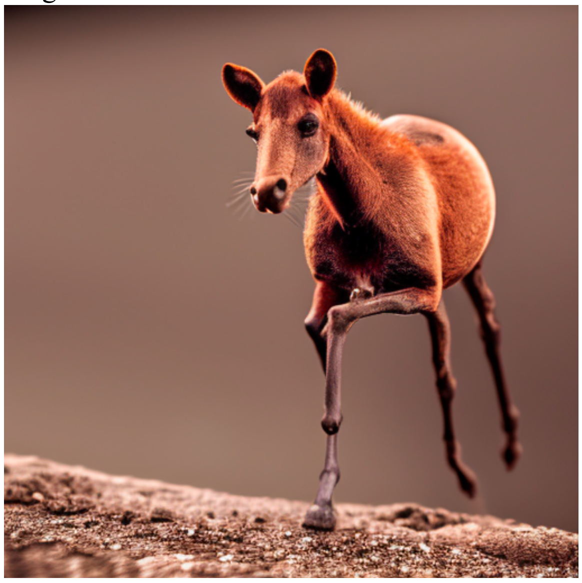


Figure 5. An image generated from 50% ant, 50% horse.

4. Discussion and use of Stable Walk

Using Stable Diffusion and other text to image models is certainly Human-AI co-creation. Using prompt engineering, users can craft prompts that guide the model towards an outcome in an iterative way. With more training, users become better at knowing what prompts lead to more desired outcomes. While it is easy to generate a generic image of a cat, to get a cat in a particular style, in a particular setting, with a particular look and feel, requires more nuance. Further, crafting prompts will only get the user so far. There are more parameters than prompts, such as cfg and step counts as we have discussed in this paper. By looking inside these algorithms we may also find a bigger parameters space than what the standard tools make available. In this work we have started looking into a UI that allow users to quickly explore cfg and steps, but also to go deeper into the space of text embeddings.

The grid view enables a quick way of exploring the effect of *cfg* and *steps* for a particular prompt. Other tools, such as *WebUI*, enable this in other ways. In WebUI you can generate a grid of images and choose parameters to vary over rows and columns, similar to in Stable Walk. This comes at a cost since you have to generate images for the entire grid. While it makes for a more exhaustive search, as it enumerates the entire grid, we find that doing this more manually (by tapping each image in a grid) in Stable Walk makes this search an active process, which aids learning. While being given a full grid of images directly might let you find a desired output quickly, it does not let you actively consider how the outputs vary with the parameters. We think making this search more explicit and an active part of the user, aids exploration beyond the current prompt.

4.1. Exploring text embeddings

One way to interpret the linear combinations of text embeddings is to consider the text embeddings as vector representations of each text prompt. These vector representations are such that prompts with similar semantic meaning will be close to each other and those with different meanings will be further apart. By taking linear combinations of text prompts, we allow ourselves to go between two or more points in this embedding space. The question is how the model will interpret more disambiguous embeddings such as one described as a point between the prompt "ant" and "horse". Such an embedding gives much more control than to simply prompt "ant horse", and allows for fine grain exploration during exploration. An example of an image generated from a text prompt between "ant" and "horse" is shown in Figure 5.

In our own explorations we have found that animals, people, and cities work particularly well. For instance the combination of cat, owl, and hedgehog, builds an expressive space of images from which you can sample (an example of which can be seen in Figure 6.). Prompts that are semantically further apart, such as an animal and a person have interesting properties. We find that there are particular points in the space where there is a sharp shift in outputs. E.g. generating images from text embeddings between "lobster" and "donald trump" mostly outputs either lobster when close to lobster, and faces when close to "donald trump", but somewhere in between there is a point around which most of the different images are rendered. This is opposed to outputs between cat and dog, which renders a more smooth transition between cat and dog.

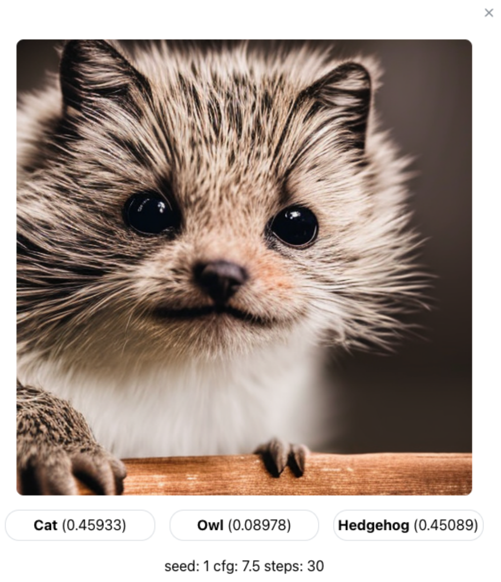


Figure 6. A combination of cat, owl, and hedgehog.

5. Future Work

While in this work we have focussed on *cfg, steps,* and text embeddings, we want to continue this work to incorporate other aspects of the diffusion process as well. E.g. can we control the output by parameterising *cfg* over each timestep in the process? In this way the user can choose a high cfg in the beginning of the process and lower it as the process comes to completion.

The same thing could be done with the weights of the text embedding. E.g. one can start the process from a prompt and then gradually move towards a different prompt by moving in text embedding space over the denoising process. We also intend to look into ways to create a UI on top of the prompt-to-prompt technique, which adds the capabilities to edit images by modifying prompts directly [13].

6. Conclusions

We have presented on-going work with a web tool that let users explore the parameter space of Stable Diffusion. There is more to text-to-image models, than simply an input prompt. Such models have inner workings that can be played around with to reach desired and creative outputs. While models are improving at a rapid pace, we believe there is still more to be learned about how users may interact with these models beyond prompts.

7. REFERENCES

[1] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour et al. "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding." *arXiv preprint arXiv:2205.11487* (2022).

[2] 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).

[3] 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*, pp. 10684-10695. 2022.

[4] Michael Cook, Jeremy Gow, Gillian Smith, and Simon Colton. "Danesh: Interactive tools for understanding procedural content generators." *IEEE Transactions on Games* (2021).

[5] Tero Karras, Samuli Laine, and Timo Aila. "A style-based generator architecture for

generative adversarial networks. arXiv e-prints." *arXiv preprint arXiv:1812.04948* (2018).

[6] Imke Grabe, Miguel González-Duque, Sebastian Risi, and Jichen Zhu. "Towards a Framework for Human-AI Interaction Patterns in Co-Creative GAN Applications." (2022).

[7] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry et al. "Learning transferable visual models from natural language supervision." In *International Conference on Machine Learning*, pp. 8748-8763. PMLR, 2021.

[8] Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. "Reproducible scaling laws for contrastive language-image learning." *arXiv preprint arXiv:2212.07143* (2022).

[9] Jonathan Ho, and Tim Salimans. "Classifier-free diffusion guidance." *arXiv preprint arXiv:2207.12598* (2022).

[10] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models." *Advances in Neural Information Processing Systems* 33 (2020): 6840-6851.

[11] Jiaming Song, Chenlin Meng, and Stefano Ermon. "Denoising diffusion implicit models." *arXiv preprint arXiv:2010.02502* (2020).

[12] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. "Elucidating the Design Space of Diffusion-Based Generative Models." *arXiv preprint arXiv:2206.00364* (2022).

[13] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. "Prompt-to-prompt image editing with cross attention control." *arXiv preprint arXiv:2208.01626* (2022).