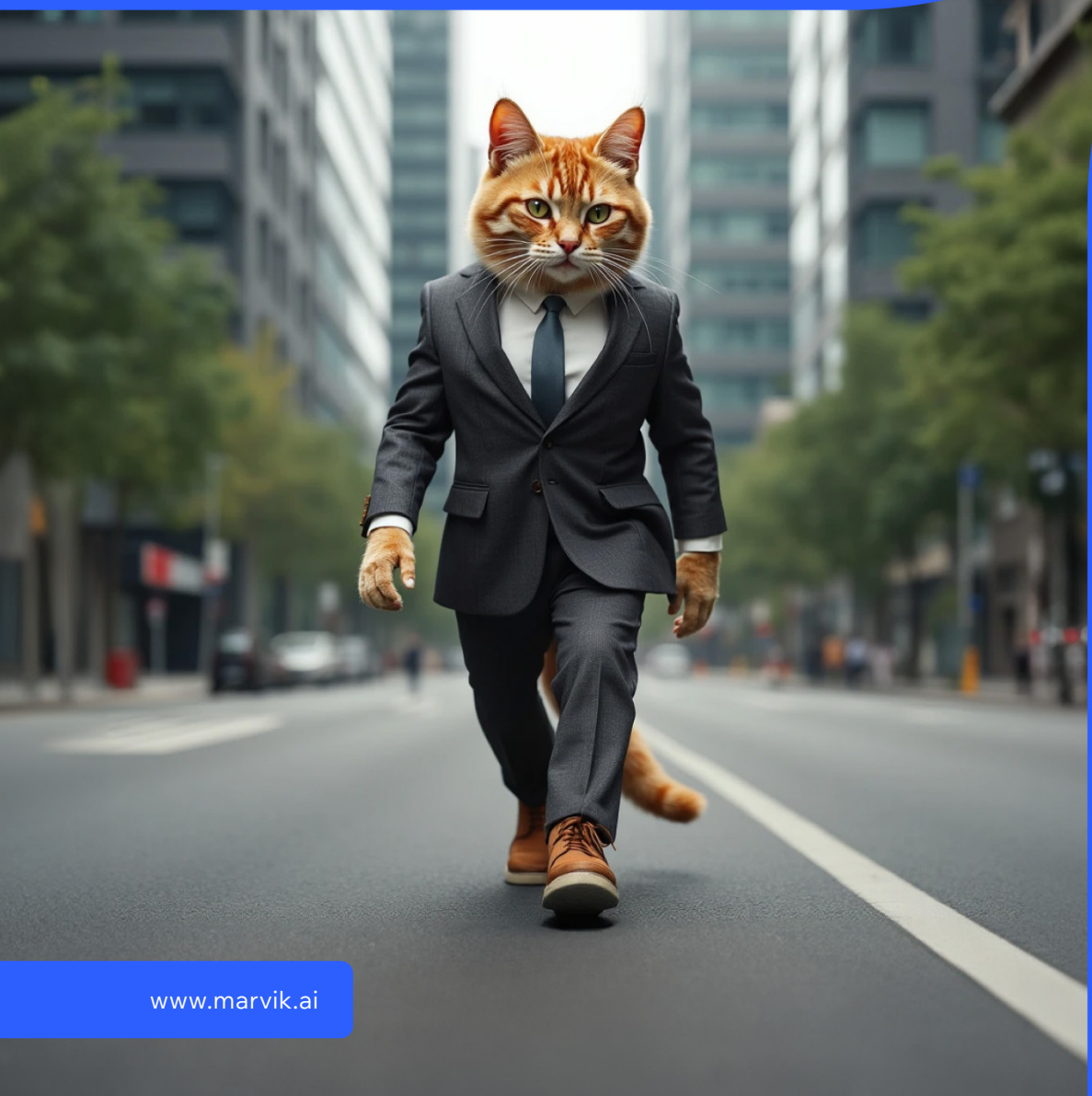
marvik. AI solutions with business impact

Mastering Stable Diffusion



www.marvik.ai

marvik.

AI solutions with business impact

Marvik is a hands-on AI consulting firm dedicated to creating impactful solutions for businesses. We excel in computer vision, natural language processing, GenAI, predictive analytics and more. Our team of +150 experts has been working for five years to deliver excellence in AI. We provide services from strategy design to full implementation, ensuring that our clients stay ahead in their industries.

Our mission is to empower businesses by harnessing data and accelerating digital transformation. We take a collaborative approach, working closely with clients to scale their internal capabilities and achieve success. At Marvik, we are committed to innovation and aligning with our clients' goals to drive results.

More information www.marvik.ai

Mastering

Stable Diffusion

Index

Introduction

Diffusion models are a type of generative models that have been shown to be able to generate high-quality images with a high degree of control. It works by starting from an image with “random noise” and removing a small amount of noise until the final image is recovered. This allows the model to generate images that are very close to the original image, but with a lot of variation.

Figure [1]: Generating image from noise.

Unlike earlier diffusion models, which operated in the “pixel space” and could be time-consuming for large images, ***Stable Diffusion takes advantage of the latent space*** (a lower-dimensional representation) ensuring much faster image generation, as we will explore in the upcoming architecture overview section.

To guide the generation process we generally use ***text prompts*** describing the image that we want to obtain, but the process of ***finding the right prompt*** to get the desired image is highly non-trivial. In this article we will review some common techniques that will help us ***control the generation process***.

Architecture overview

In this section we will be reviewing the architecture of the Stable Diffusion model. But first we will review the elementary concepts that are necessary for understanding it.

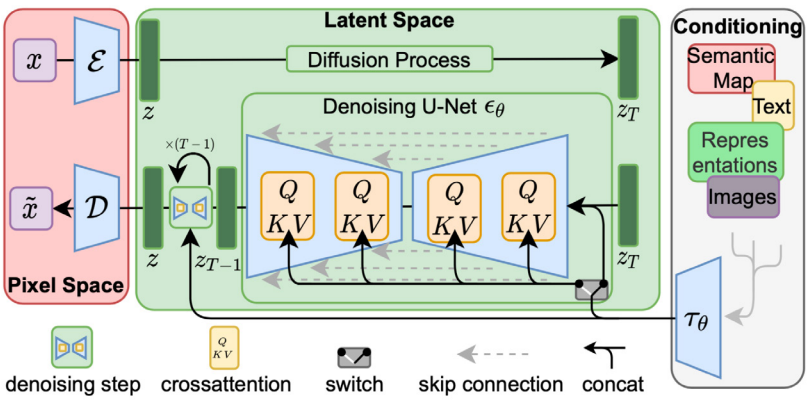


Figure [2]: Stable Diffusion Architectures - Bombach & Blattmann, et al. 2022.

Diffusion process in diffusion models

The diffusion process can be divided into two parts: The ***forward diffusion*** and the ***backward diffusion***.

In the forward diffusion we gradually add noise to the image, the amount of noise that is added at each step is controlled by a scheduler/sampler. In the next image we can see how we gradually add noise to the original image until it is indistinguishable from pure noise (at least for our eyes).

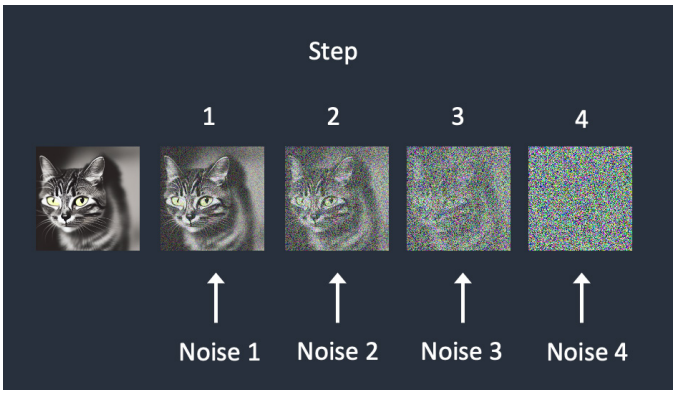


Figure [3]: Forward diffusion process.

Backward diffusion consists of ***predicting the amount of noise*** of a given image and extracting a bit of it. The more steps we take to remove all the noise, the higher the image quality will be. In this image we can see how we gradually subtract a bit of noise from the image and the result becomes clearer at each step. This process is also controlled by the scheduler, how much noise is removed from the image at each step and how many steps we do. For a more in depth review of how schedulers work and which options are out there check this blog.

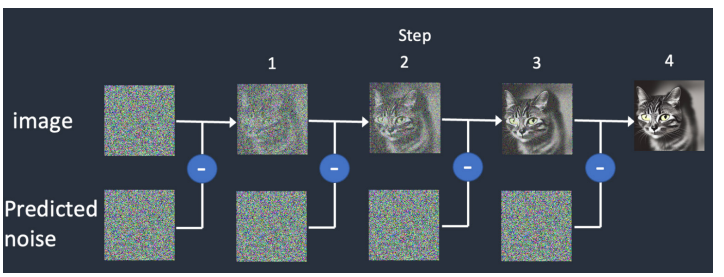


Figure [4]: Backward diffusion process.

In all the examples we have seen so far we added noise to the original image, we were working in what we call the “pixel space”, and in the beginning all diffusion models worked in this space (some still do, Imagen and Dall-E 2 to name a few). But this is very slow, a faster alternative was proposed in the 2021 paper High-Resolution Image Synthesis with Latent Diffusion Models, they showed that we can achieve high quality generation of images using the “latent space” instead of the pixel space for the diffusion process.

Variational Autoencoders (VAE) in Diffusion Models

For this they use a model called Variational Autoencoder (VAE) to ***compress the image*** into a much smaller ***vector***, and then do the diffusion process with the new compressed vector instead of the original image.

A variational autoencoder is composed of two parts: an ***encoder***, that compresses the image into the ***latent space***, and a ***decoder***, that ***recovers*** the image from the corresponding latent space vector.

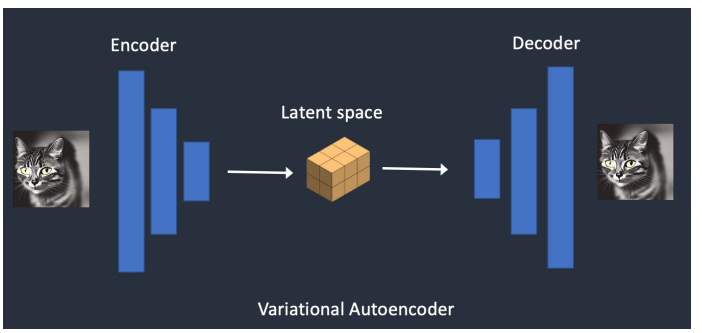


Figure [5]: VAE Architecture

This process is not perfect, it always loses some information, but it is ***almost indistinguishable*** to the naked eye, as you can see in the next example.

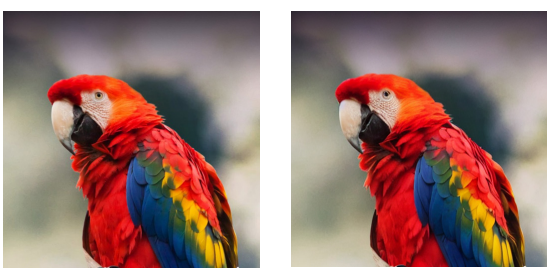


Figure [6]: Which is the real image?

In the diagram we marked the ***encoder as*** $\mathcal{E}$ ***the decoder as*** $\mathcal{D}$, the input image as x and the corresponding vector in the latent space as z. As you can see, the only time we work in pixel space is before encoding the input image, and after decoding the output vector. The whole diffusion process occurs in the latent space.

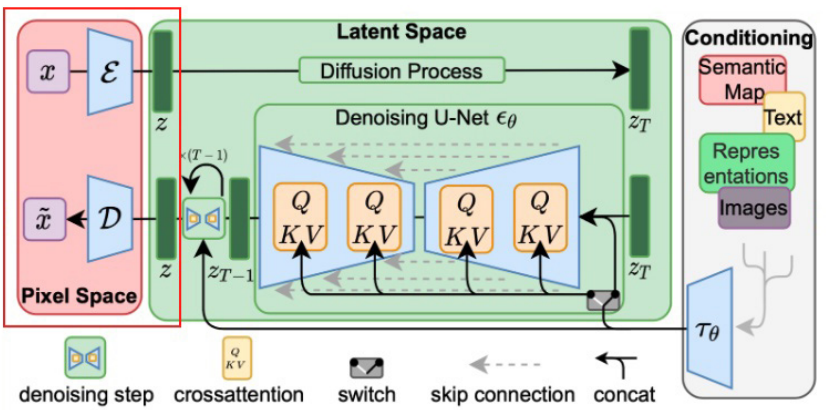


Figure [2]: Stable Diffusion Architectures - Bombach & Blattmann, et al. 2022.

Classifier Guidance

One of the first methods proposed to control the diffusion process is called classifier guidance and was introduced in the paper [Diffusion Models Beat GANs on Image Synthesis](). The idea is that we can use the output of a classifier to "guide" the diffusion process in the direction that we want, doing this we ***lose variability in the outputs*** that we can generate, but ***increase the fidelity*** of the image to the target class. The strength of this guidance could be controlled by moving the ***"guidance-scale" parameter*** during sampling. The drawback of this method is that we need to ***train a classifier for the specific classes*** that we want to generate.

Text Conditioning

Probably the most common method of controlling the diffusion process today is text-conditioning, using a text-based prompt to guide the generation of the kind of images that we want. For example, this are the images that Stable Diffusion generated with the following prompt:

Prompt:

```
Beginning of the universe the big bang, amazing color palette, clean
lines and shapes, illustration, trending on artstation, small de-
tails, black background, clean, 4k, hd,  artgerm
```

Something that you might have noticed is that we created four very different images from the same text prompt. So, how did we do that? When we are doing inference we take a vector in the latent space filled with ***"random" noise***, so depending on how we generated that random noise we will get different images, this can be set by a parameter called "seed" or ***"random seed"***.

How does stable diffusion work?

Stable Diffusion V.1.5 uses a model called ***CLIP as a text encoder***, i.e. for transforming the text prompts into vectors that can be fed to the ***attention Unet*** (more on how this works in the next section). These vectors are fed into the Unet that contains blocks with self-attention mechanism (borrowed from the popular transformer architecture), these blocks provide the network with information about the text-prompt and enable it to use it to guide the reverse diffusion process and recover an image that's hopefully similar to the one intended.

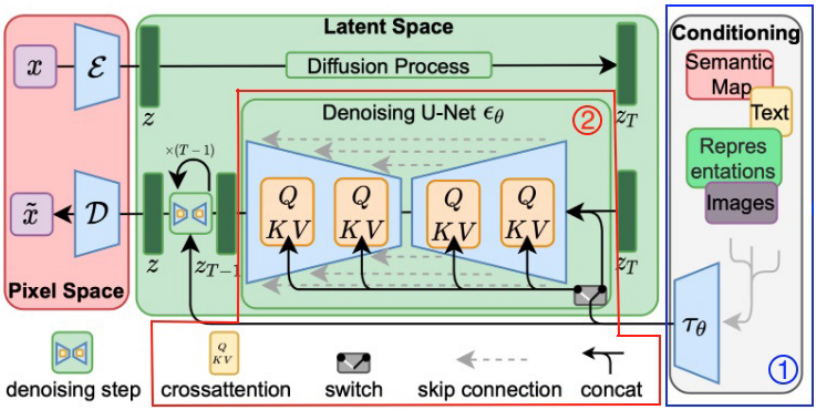


Figure [2]: Stable Diffusion Architectures - Bombach & Blattmann, et al. 2022.

As we can see in **(1)** in the diagram, we can ***condition the network with many different inputs***, including semantic maps, images and, of course, text. Again in the diagram, **(2)** represents the Unet with its attention blocks, it shows how the encoded text is fed to each of the blocks in turn. Also, we have the "switch" which means that we can ***turn on/off the conditioning*** for a particular image during training. This will be helpful for classifier-free guidance, a controlling technique inspired by classifier guidance, but that ***doesn't need a pre-trained classifier.***

Classifier-free guidance

Classifier-free guidance or CFG, is an interesting trick that was introduced in this paper (Classifier Free Diffusion Guidance). The authors propose to use image captions and train a conditional diffusion model, putting the classifier part as conditioning for the noise-predictor UNet, instead of training a separate classifier and using class labels.

During training this is achieved by randomly replacing the prompt for generating the target image with an ***empty prompt "" a percentage of the time*** (A hyperparameter that can be changed). At inference time we have at our disposal a parameter called CFG scale that controls how much importance we give to the prompt while generating the image. When we set this value to 0, the prompt is ignored. A higher value steers the diffusion process towards the prompt.

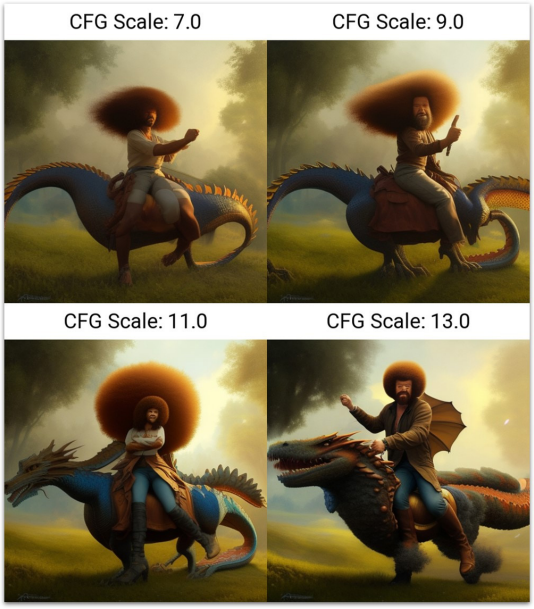


Figure [7]: “Bob Ross riding a dragon”.

In these examples we are trying to generate a picture of “Bob Ross riding a dragon” and we are varying the classifier free guidance parameter. Scale 7 is the default value representing giving a balanced weight to the prompt. But in these examples we can see that the CFG scale has to be at least 11 to get results that resemble what we are looking for.

Common tools to guide the diffusion process

Text to image methods

Negative prompting

When we talked about CFG we mentioned that at inference time we could control how much impact the prompt had compared to generating with an empty prompt. Well, there is another possibility here, instead of using an empty prompt for this, we could use what is called a “negative prompt”, where we put all that we DON’T want in our image. For a more detailed explanation of how this works at the code level review this wiki.

This technique can be used for a lot of things, for example removing things that we don’t want in the image (following example), changing the style (anime, realistic, etc) and even improve anatomy if we are trying to generate people (extra fingers, bad anatomy, etc).

Left: Generated image with people.
Right: Generated Image without people (negative prompt).

On the left we have an example of an image generated using Stable Diffusion, it was generated with this prompt: autumn in paris, ornate, beautiful, atmosphere, vibe, mist, smoke, fire, chimney, rain, wet, pristine, puddles, melting, dripping, snow, creek, lush, ice, bridge, forest, roses, flowers, by stanley artgerm lau, greg rutkowski, thomas kindkade, alphonse mucha, loish, norman rockwell

But what if we didn't want people in the image? Well we can add people as a negative prompt and try again. On the right the people disappeared, amazing! But be warned that other details may change (even with all the other parameters maintained), as with the building that was on the right that has disappeared.

CLIP interrogator

This technique was introduced in this repo, given an image it outputs a prompt that's likely to generate a similar image. This is done using lists of words, for example, artist names, "flavors" and "mediums" and combining them to ***generate a prompt***, then these prompts are fed to CLIP along with the image, looking for the combination with the highest score.

As we are using ***fixed lists of words***, what we can obtain is limited to the possible combinations with the given words. One thing you can do is use your own lists of candidate words to improve the results if you work with images from a particular domain. The base model could be changed also, for example if you work in the fashion industry, a model like FashionCLIP would be better suited for this task. a

Figure [8]: Gandalf in the series of "The lord of the rings".

Extracted prompt:

```
Man with a long beard and a long beard holding a sword and a staff
in front of a castle, a detailed matte painting, Edward Otho Cresap
Ord, II, antipodeans, weta digital
```

As you can see, the ***generated text is sometimes repetitive***, but it provides us with some information about the picture. If we try to regenerate the image from the prompt this is what we could get:

It has some of the flavor of the original image, but if we want to get something closer to the original we will have to manually correct the prompt, removing the repetition and some superfluous words that appeared.

Prompt:

```
Man with a long beard holding a sword and a staff, standing in front
of a castle, a detailed matte painting, antipodeans, weta digital
```

Textual inversion

Usually, text prompts are tokenized into an embedding before being passed to a generator model. Textual Inversion does something similar, but instead of creating an embedding from a prompt, it ***learns a new token embedding***, v*, from a special token S* as in the diagram below. This new token v* is used to guide the generation, which helps the diffusion model understand ***new concepts from just a few example images***.

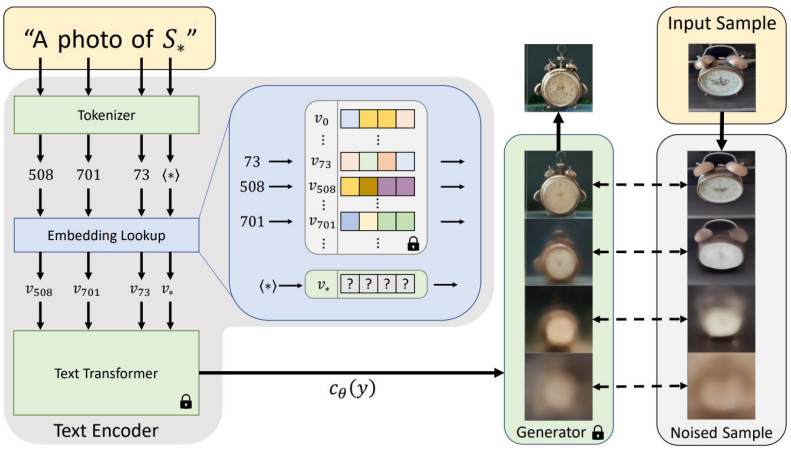


Figure [9]: Architecture overview from the Textual Inversion.
Reference articles: https://huggingface.co/docs/diffusers/training/text_inversion

To do this, textual inversion uses a freezed generator model (its parameters are not updated) and noisy versions of the training images of the target concept. The generator then tries to predict the original images, and the new embedding v* is optimized based on how well the generator does. At the end of a few thousand steps, we get a new embedding v* that hopefully encapsulates enough information of our target concept, and this embedding is associated with the special token S* that we can use in our prompts to guide the diffusion process to generate images using the concept/style that we trained it for.

Checkpoints

There is a big community around generating images with Stable Diffusion, with many finetuned models available, for example Civitai has a big variety of them. Using a model that has been ***finetuned with images similar to the ones that we are trying to generate*** is generally a good starting point for improving the results in a particular domain.

Checkpoint

Stable Diffusion V1.5

Prompt:

```
Masterpiece, best quality, Day of the Dead, Dia de los Muertos,
skull, sugar skull, calavera, Mexican culture, traditional, celebra-
tion, remembrance, colorful, festive, face paint, flowers, marigolds,
altar, offerings, candles, skeleton, afterlife, spirits, ancestors,
Mexican holiday, vibrant, decorative, traditional dress, papel picado,
symbolic, festive makeup, cultural heritage, joyful, commemoration,
artistic, folk art, spiritual, lively, macabre, death, festive atmo-
sphere, family, community, lively celebration, festive music.
```

Negative prompt:

```
Asian, (worst quality, low quality:1.4), watermark, signature
```

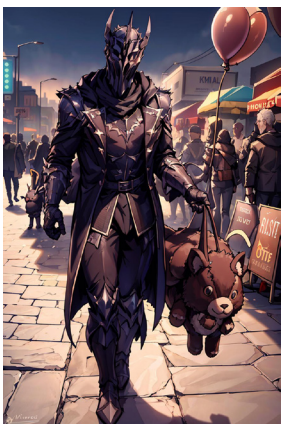
As you can see from the images, even with the same prompt the difference between the ones generated using the checkpoint and SD is huge. This particular checkpoint is finetuned so that almost any image produced with it is done in a Disney Pixar cartoon style (Checkpoint images and prompts come from the model card examples).

LoRA

Low-Rank Adaptation of Large Language Models (LoRA) is a technique developed by Microsoft that ***trains only a small subset of the weights*** of the model while maintaining the rest frozen. This technique was first implemented in the context of diffusion models here.

There the author proposed to only train the attention weights from the Unet using this technique to learn a particular object or style without having to retrain the whole model. The main advantages of this are the size of the new weights, if you have the base model you only need the changes in the weights which are generally really small (2-100 MBs depending on the dimension, a hyperparameter that can be set during training), and the training time, this technique is ***really fast compared to fine tuning the whole model***, as most of weights are kept frozen.

Below we can see images created with this LoRA trained with Sauron images:



LoRa + Model

Only Model

Prompt:

```
Best quality, highres, high resolution, high detail, sharp focus, cinematic shadow, directional lighting,  Lord Sauron wearing a letterman jacket, walking through a bright colorful festival, holding a fluffy stuffed animal and the string of a floating balloon.
```

In this case we are using two different checkpoints plus a LoRA particularly trained with Sauron images (LoRA images and prompts come from the model card examples). The images generated using the LoRA weights are really good at generating this particular character, while the images generated using only the base model (but same prompt) can't generate what we want, even though Sauron was specified in the prompt.

Image to image methods

img2img

This technique was introduced in SDEdit (Guided Images Synthesis And Editing With Stochastic Differential Equations) and is adapted from the way that we train diffusion models: given an input image, we gradually add noise to it until it loses some or almost all its particularities and then we use reverse diffusion to try to recover the original image.

The amount of noise that we add is determined by a parameter sometimes called the noise strength, and it controls the flexibility given to the model to modify the output. This parameter goes from 0 to 1, with zero strength returns basically the original image, the values in between produce more changes to the original image as we move farther from zero, and one returns an almost completely different image.

Figure [10]: Doodle Img2Img result from workflow.
Reference blog: https://mccormickml.com/2022/12/06/how-img2img-works/

Inpainting

What do you do if you manage to create the perfect portrait to find out that the person in the image has three arms or six fingers? One option would be to open the resulting image in photoshop and start modifying the image according to our needs. Another option, for those of us that are lazy or not very good at photoshop, is to use a technique called inpainting, another “img2img” method. In this case we will use the image we obtained as input and add a ***mask over the region of the image that we want to change***, in this case the extra arm or finger. The rest will be exactly the same as the previous method, but it will have effect only over the masked portion of the image.

The fidelity of this modified part to the original one will be controlled again by a parameter between 0 and 1, in this case called “inpainting strength”. There is also another important parameter, the masked content, which lets us define where the noise we add to the masked part comes from. We can select original if we want to create something similar to the original, for example changing a bit the facial features of our model, or latent noise/latent nothing if we want to generate something from scratch, as we would probably use in this case to remove the extra arm/finger.

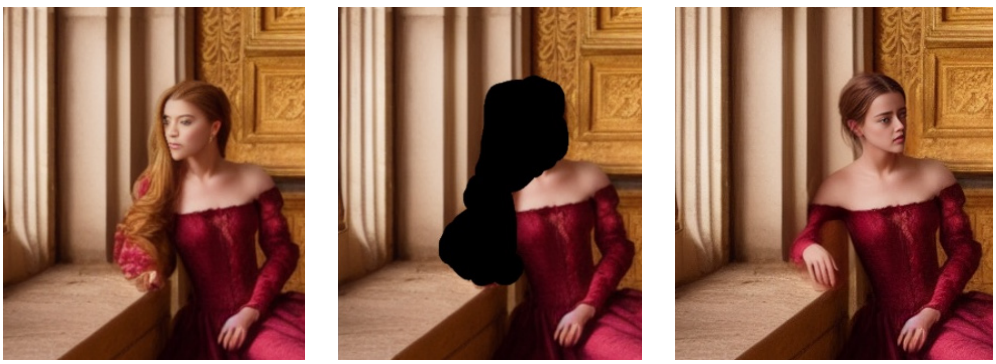


Figure [11]: Inpainting human faces.
Reference blog: https://stable-diffusion-art.com/inpainting_basics/

In this case the image generated by Stable Diffusion (the first one) had a few details, the right hand and the face didn’t come out well. To fix this we are masking a part of the image (black in the second image) and using ***inpainting over that area to fix*** that, and in the third image we have the result, much better right?

These examples were made using the Stable-Diffusion-webui tool, which lets you run Stable Diffusion locally in your computer and has a myriad of tools available, including img2img and inpainting.

ControlNet

As we have seen, with img2img methods we have much more control over the generation without relying too much on getting the perfect prompt. But these methods also have their disadvantages, it is hard to ***influence the generation*** only in some particular aspects of the input image, be it ***colors, pose, shape*** or others. To get more control over the generation we can use a tool called ControlNet. This lets us control the generation process by ***providing an image for conditioning***

the output. There are many conditioning possibilities that we could use, a few examples include canny edge, normal map, depth map, HED map and even user-made scribbles.

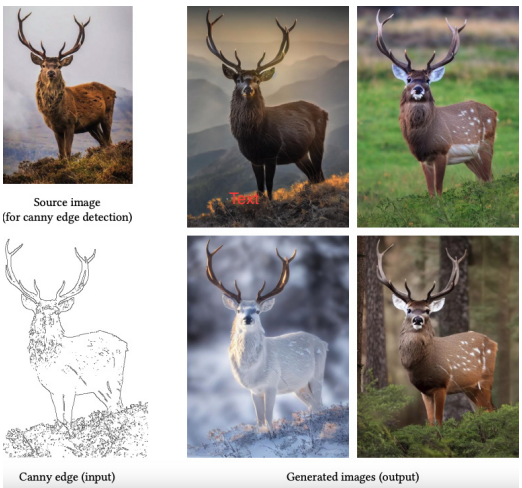


Figure [12]: Example of controNet with canny edge.
Reference repository: https://github.com/lllyasviel/ControlNet

Upsampling

By default methods like Stable Diffusion generate images of 512x512 pixels, but what if we want to generate images of higher quality?

Traditional algorithms for resizing images, such as nearest neighbor interpolation and Lanczos work by taking into account the pixels of the original image, then enlarge the canvas and fill the new pixels by performing mathematical operations in the pixel's of the original image. In the past few years many methods that utilize AI to upscale images have appeared, the most commonly used are based on GANs (Generative Adversarial Networks). How do these models work?

Well, first good-quality images are artificially corrupted to emulate real-worl degradation, and then are reduced to smaller size, and then a neural network is trained to recover the original image from the modified one.

Figure [13]: Upsampling images with high quality.
Reference blog: https://stable-diffusion-art.com/ai-upscaler/

Image prompt adapter

“An image is worth a thousand words”: as a more detailed prompt will generate better results, an image along with image prompt adapter (or IP-Adapter) can be used to generate something related to a source image.

IP-adapter is very useful in tasks such as preserving a person’s identity in the generation. It was trained in a similar way to CLIP, by pairing images and text, so when we use it it takes the input image and conditions the generation closely with it.There is an IP-adapter that corresponds to the Stable diffusion model used, and there are the “regular”, full-face, plus, plus-face, face-id, and face-id-plus. The current best model to preserve a person’s identity is the ip-adapter face-id plus.

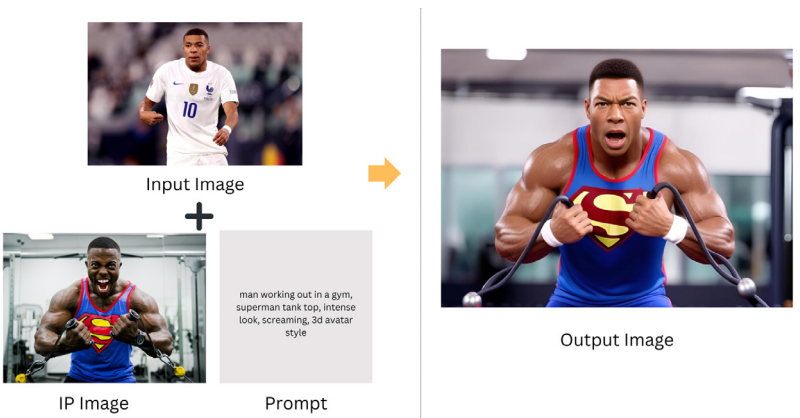


Figure [14]: Upsampling images with high quality.
Reference blog: https://blog.segmind.com/ip-adapter-xl-models-canny-depth-pose

Novel models

FLUX

What is FLUX?

Flux is a new AI model created by Black Forest Labs, designed to generate images with remarkable quality and prompt fidelity. It's built using a combination of transformer and diffusion techniques along with 12 billion parameters, making it perform excellent in prompt following, visual quality, output diversity, typography, and size/aspect variability compared to other current state of the art models such as midjourney-v6.0.

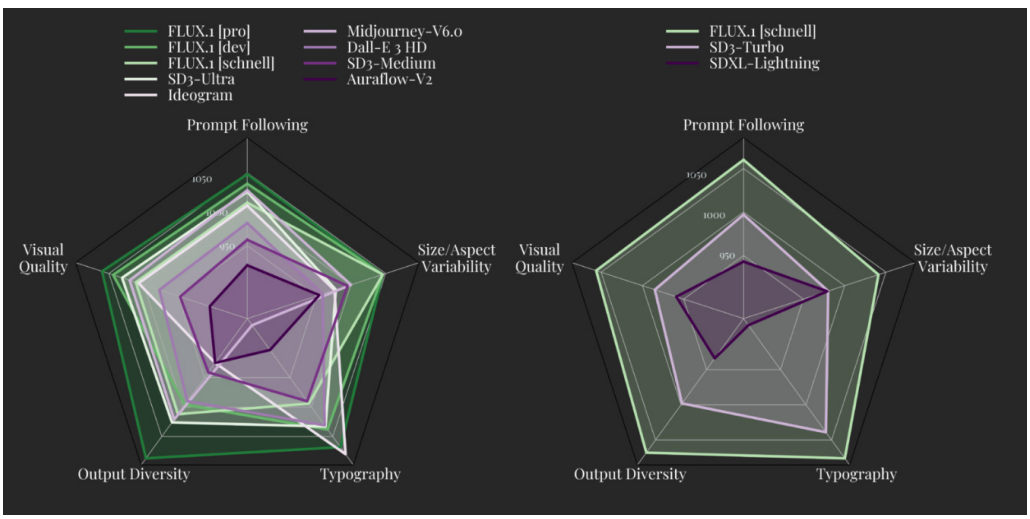


Figure [15]: Comparison between FLUX and other state of the art models.

Who created FLUX?

Black Forest Labs was founded by the creators of Stable Diffusion models such as: Stable Diffusion models for image and video generation (Stable Diffusion XL, Stable Video Diffusion, Rectified Flow Transformers), and Adversarial Diffusion Distillation.

How does it work?

The model does not have a specific research paper associated with it, but according to the announcement “All public FLUX.1 models are based on a hybrid architecture of multimodal and parallel diffusion transformer blocks and scaled to 12B parameters.” The quoted papers suggest that FLUX uses multimodal transformer-based backbones such as MM-DiT, along with Rectified Flows, which allows it to reduce the sampling steps without losing much performance.

Versions

The model was presented in 3 different variants:

- **FLUX.1 [pro]:** Best performance, available only through API.
- **FLUX.1 [dev]:** open-weight, with non commercial license. It’s derived from the pro version, and has almost the same quality.
- **FLUX.1 [schnell]:** The fastest model, open-weight and has Apache2.0 license.

Flux is available in Github and Diffusers, and is also available in platforms like fal.ai.

Text to image

The area where this model outperformed the others is in the text preservation. In Figure [16], we can see that even though the text “marvik.” was preserved for both FLUX and Dall-e, the FLUX generations have a more smooth look.

Prompt:

```
A cup of coffee that has the word “marvik”, in a table in the grand canyon.
```

FLUX

DALL-E 3

Figure [16]: Examples using Dall-e 3 and Flux.

When the text is longer, the comparison becomes clearer:

Figure [17]: FLUX.1 [schnell] generation. Prompt: A laptop in the screen “marvik we are a hands-on machine learning consulting firm”, in a table in the grand canyon.

Figure [18]: Dall-e 3 generation. Prompt: A laptop in the screen “marvik we are a hands-on machine learning consulting firm”, in a table in the grand canyon.

Image to image

FLUX also performs excellent in image-to-image generation tasks. In Figure [19], we used the image of a couch to animate how it would look with kids on it.

Figure [19]: Image-to-image generation using FLUX.1 [dev].
Prompt: A couch with two kids playing.

Tools Comparison

There is no “best” tool out there that will solve any given problem (at least for now). Almost all the tools that we have seen in this article have different purposes. For example, if you want to modify small details from an image, inpainting is probably the way to go. But if you don’t mind keeping most of the original picture and just want to create a new image following a certain structure, then img2img might be just what you need.

Final thoughts

We are at a point where generative models like Stable Diffusion can create incredible pieces of art, realistic photos and almost anything else you can imagine. But even with all the controlling tools that we saw here, it is really hard to predict how the output will turn out, for example you will get images with artifacts or persons with bad anatomy from time to time. This lack of reliability makes it really hard to put these models in a production pipeline. If you want to put them in production, you will need to implement lots of sanity checks to prevent mistakes. Another thing that you should take into account is that these models come with huge biases in their training data, which has been scraped from all over the internet.

References

1. https://onceuponanalgorithm.org/guide-what-is-a-stable-diffusion-seed-and-how-to-use-it/
2. https://arxiv.org/abs/2112.1075
3. https://stable-diffusion-art.com/how-stable-diffusion-work/
4. https://stable-diffusion-art.com/how-stable-diffusion-work/
5. https://stable-diffusion-art.com/how-stable-diffusion-work/
6. https://github.com/fastai/diffusion-nbs/blob/master/Stable%20Diffusion%20Deep%20Dive.ipynb
7. https://mccormickml.com/2023/02/20/classifier-free-guidance-scale/
8. https://www.fotogramas.es/series-tv-noticias/a19454229/serie-el-senor-de-los-anillos-ian-mackellen-gandalf-entrevista/
9. https://textual-inversion.github.io
10. https://www.reddit.com/r/StableDiffusion/comments/xiwkhy/here_is_an_example_of_my_img2img_with_stable/
11. https://stable-diffusion-art.com/inpainting_basics/
12. https://arxiv.org/pdf/2302.05543.pdf
13. https://stable-diffusion-art.com/ai-upscaler/
14. https://blog.segmind.com/ip-adapter-xl-models-canny-depth-pose/
15. https://blackforestlabs.ai/announcing-black-forest-labs/

marvik.

AI solutions with business impact

Got a project? growth@marvik.ai

More information www.marvik.ai