Stable Diffusion Explained, Really

Roger Lam
lamroger.com

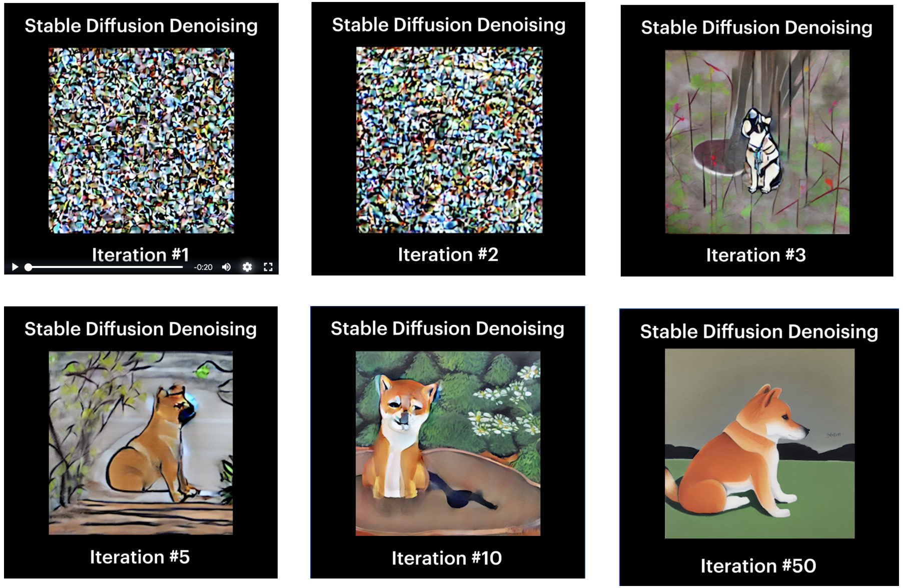
Stable Diffusion is a latent diffusion model (LDM)

Stable Diffusion (SD) uses the concept of “diffusion” to generate images based on the input prompt.

Think of how ink diffuses in water.

It progressively moves to a stable state.

SD works in the same way. But it’s going from randomness to a better and better image.

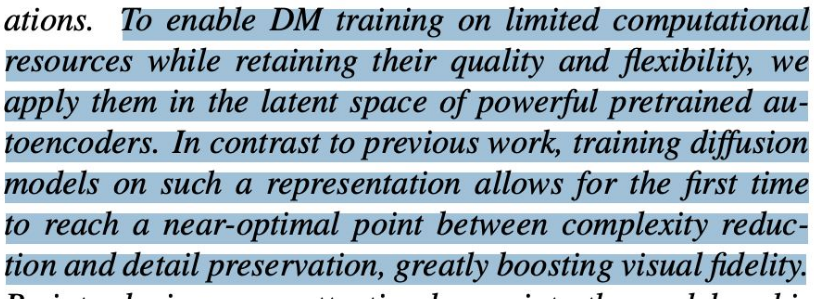

Stable Diffusion Denoising
Iteration #1
-0:20
Stable Diffusion Denoising
Iteration #2
Stable Diffusion Denoising
Iteration #3
Stable Diffusion Denoising
Iteration #5
Stable Diffusion Denoising
Iteration #10
Stable Diffusion Denoising
Iteration #50

https://www.graphcore.ai/posts/how-to-run-stable-diffusion-inference-on-ipus-with-paperspace

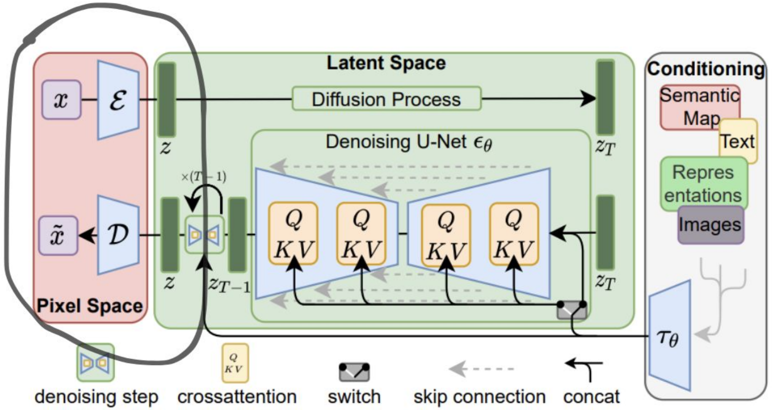
The latent space allows higher quality for lower compute

While LDMs weren't the first to generate images, it's use of a latent space gave state of the art improvements.

Think of a 1024 x 1024 pixel image. That's ~1,000,000 pixels. By using an encoder to map it into a smaller vector, we compress the image, distilling the information, and speed up our diffusion process.



ations. To enable DM training on limited computational resources while retaining their quality and flexibility, we apply them in the latent space of powerful pretrained autoencoders. In contrast to previous work, training diffusion models on such a representation allows for the first time to reach a near-optimal point between complexity reduction and detail preservation, greatly boosting visual fidelity.



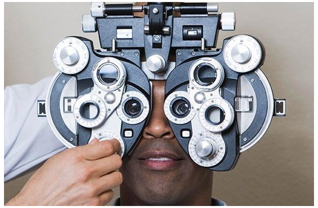
We train the model by adding noise to images

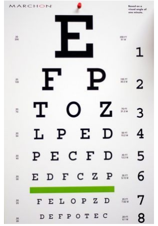
Think of an eye exam.

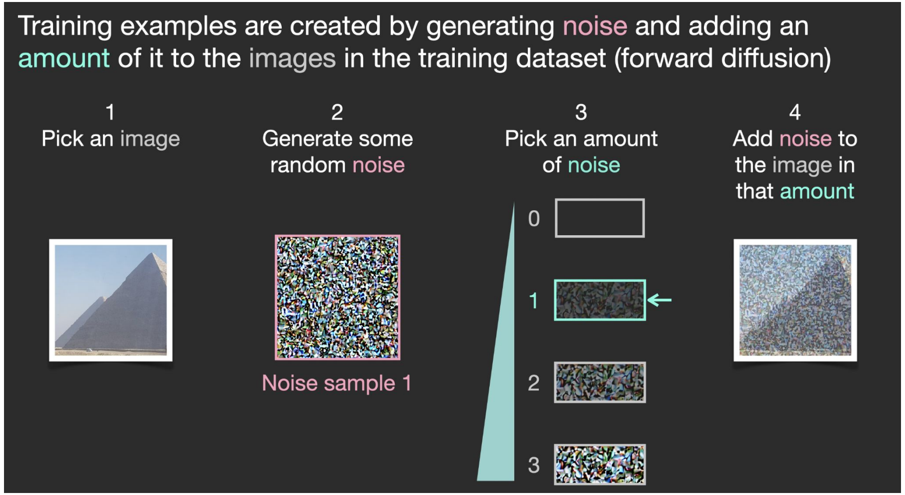
We have a letter E. We can apply a lens at various degrees of blurriness to generate blurry E’s.

We also know how to undo it because we know the degree of blurriness we added.

This is how we simulate data of the reverse of the deblurring process.







Training examples are created by generating noise and adding an amount of it to the images in the training dataset (forward diffusion)

1 Pick an image

2 Generate some random noise

3 Pick an amount of noise

4 Add noise to the image in that amount

https://jalammar.github.io/illustrated-stable-diffusion/

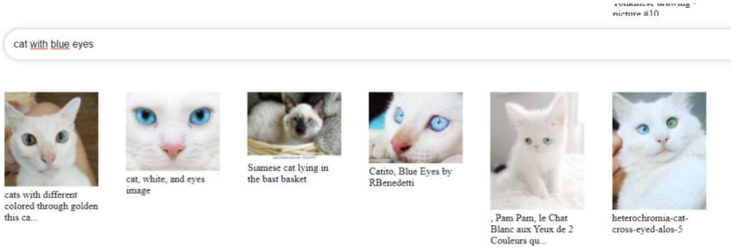
We train on images with text labels

To go from text to image, we also need labels for the images.

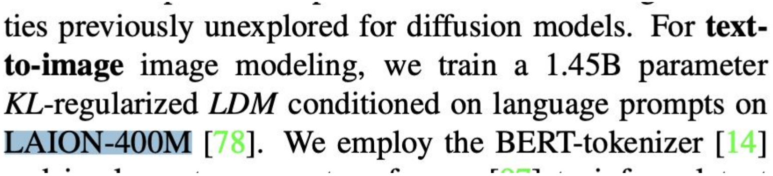
SD used LAION-400M, an open dataset of 400M english text-image pairs.

The paper uses BERT to convert the text into the latent space but we can use any tokenizer.

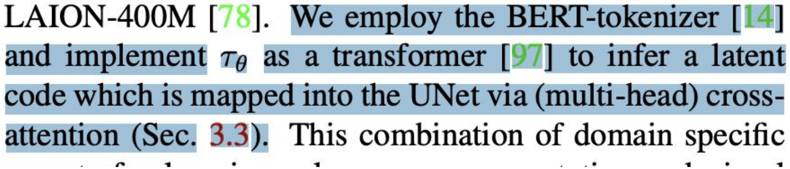
Next, we need to incorporate it into our model.


picture #10
cat with blue eyes
cats with different colored through golden this ca...
cat, white, and eyes image
Siamese cat lying in the bast basket
Catito, Blue Eyes by RBenedetti
, Pam Pam, le Chat Blanc aux Yeux de 2 Couleurs qu...
heterochromia-cat-cross-eyed-alos-5

https://arxiv.org/pdf/2111.02114.pdf



ties previously unexplored for diffusion models. For **text-to-image** image modeling, we train a 1.45B parameter *KL*-regularized *LDM* conditioned on language prompts on LAION-400M [78]. We employ the BERT-tokenizer [14]



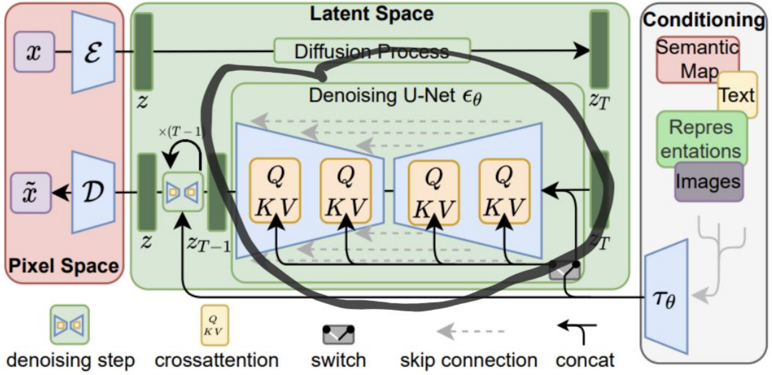
LAION-400M [78]. We employ the BERT-tokenizer [14] and implement τ_θ as a transformer [97] to infer a latent code which is mapped into the UNet via (multi-head) cross-attention (Sec. 3.3). This combination of domain specific

Just stick the text embedding in

We append attention layers to our diffusion process.

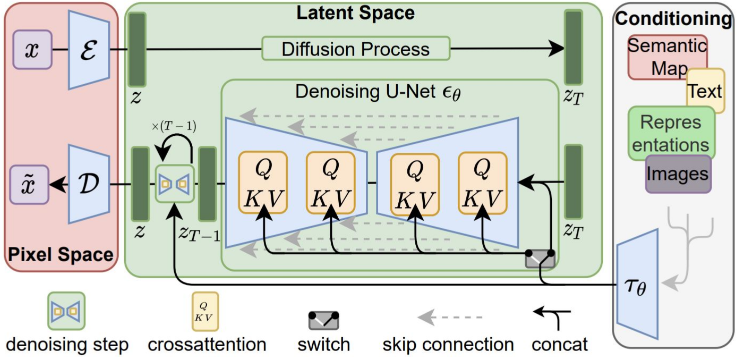
Since the processing is done multiple times in each step (1 step = 1 black circle), appending is effective because the next calculation will now take the new attended value into consideration.

Now, denoising responds to text.



And that’s pretty much it!

We have a model that can convert text (and other things) to embeddings, pass the embedding to the attention mechanism to be used, then start randomly and progressively get a better representation in the latent space, and then decode it into a pixel image!



Thank you!

Hope this was helpful!

For further learning:

- Illustrated Stable Diffusion by Jay Alammar
- Practical Deep Learning: Part 2 by Jeremy Howard

https://stability.ai/news/stable-diffusion-3