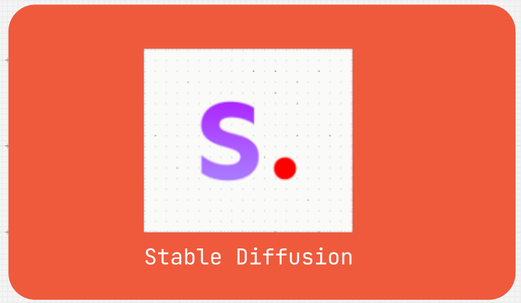
100xEngineers

Lecture 2

PlaygroundAI and How Diffusion Works

What Stable Diffusion does

“cat wearing a suit sitting in an office”



Diffusion Models

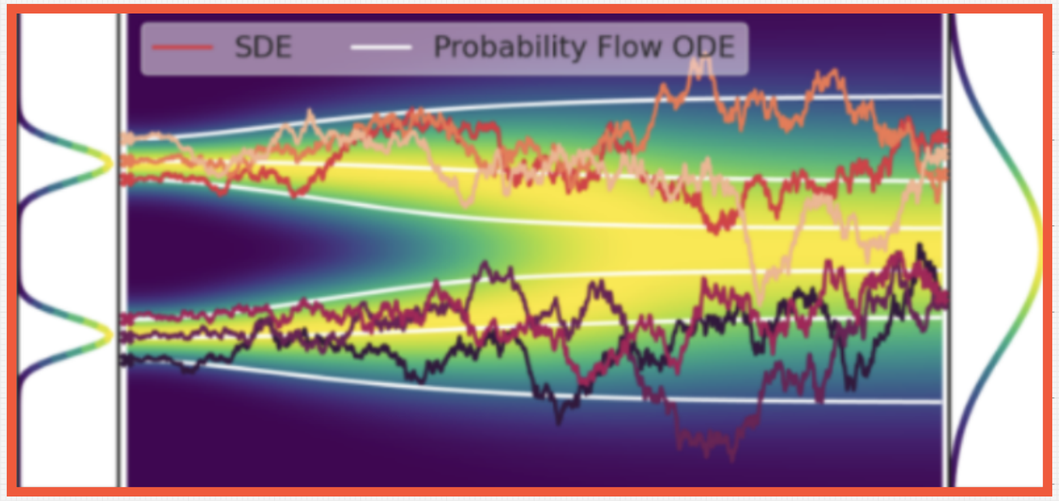
Forward and Reverse Diffusion

Fixed forward diffusion process

Generative reverse denoising process

Training

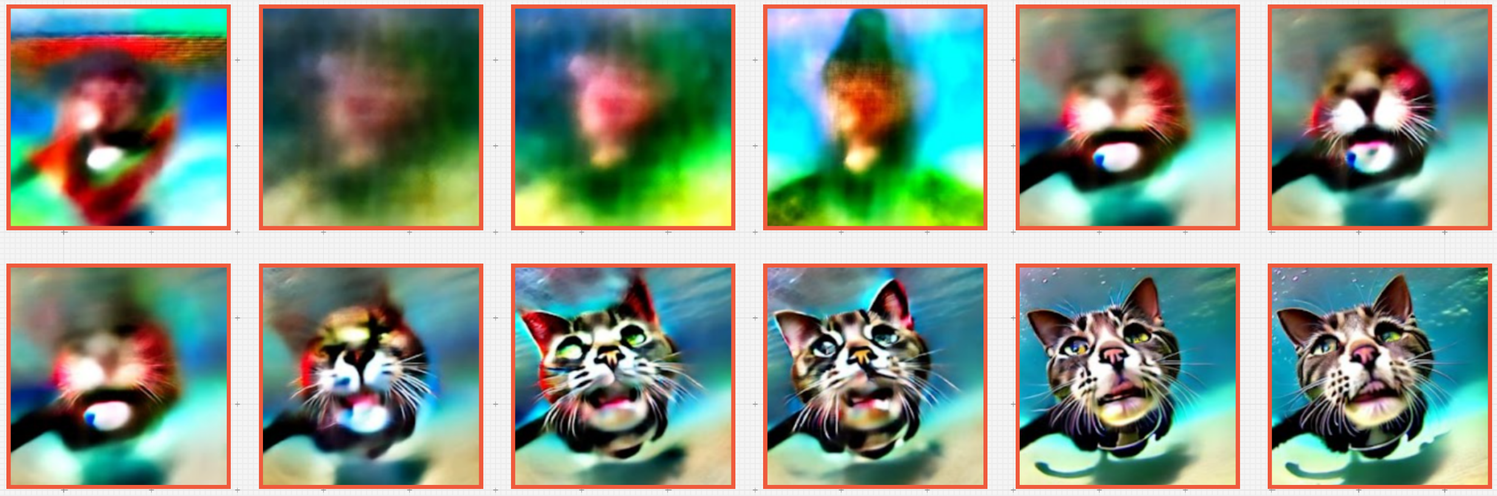
Fixed forward diffusion process



Where the **name comes** from

Reverse Diffusion

Start from a noisy image and recover a cat or a dog.



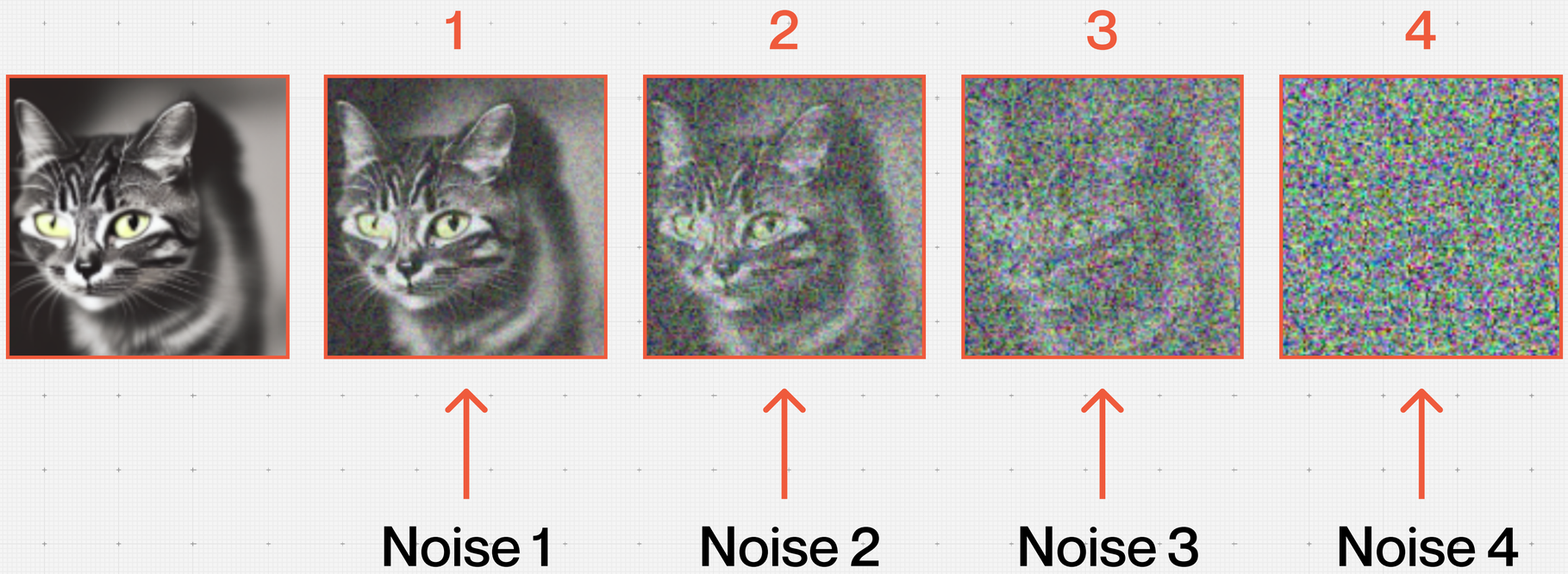
What we have so far

- For reverse diffusion, we need to know how much noise to subtract from initial noisy image.
- We need a Noise-predictor
- Train a neural network model to predict the noise

How Training is done

- Choose an image, say cat.
- Generate some random noise
- Corrupt the cat image by gradually adding noise to it step-by-step
- The model eventually learns how much noise it takes from cat image to complete noise image. It learns to predict based on multiple trainings.
- Achieved by showing it the right answer (supervised learning) and weight tuning.

Noise Predictor Training



How Reverse Diffusion works

- Generate a random noisy image
- Use the trained noise-predictor to tell us the amount of noise to subtract from the initial image
- Repeat the steps multiple times
- We arrive at the cat

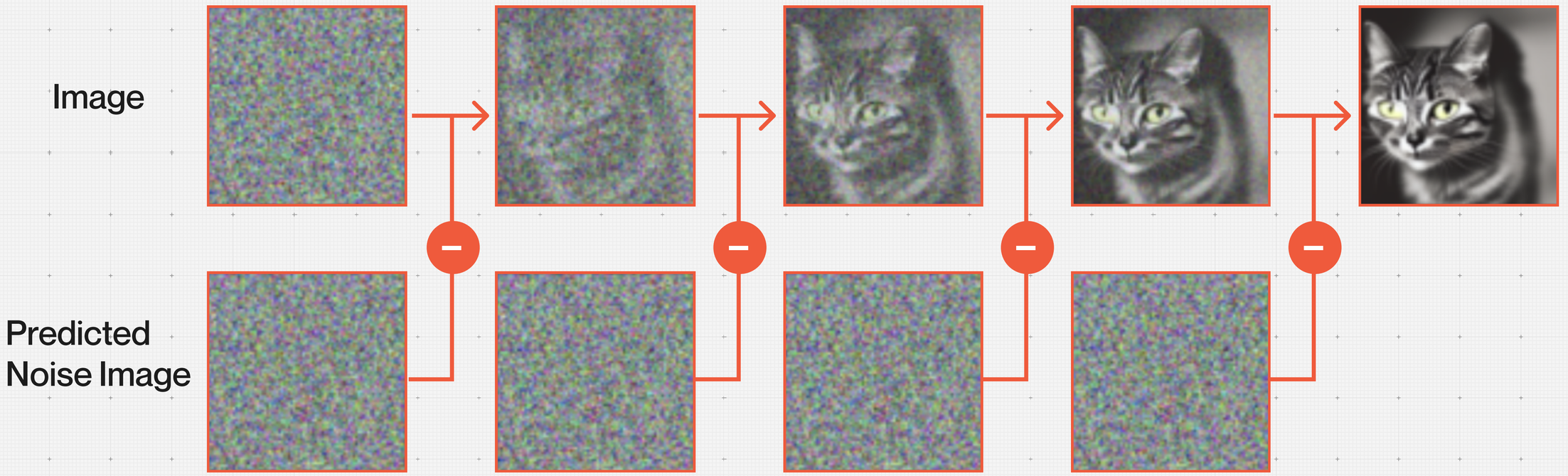
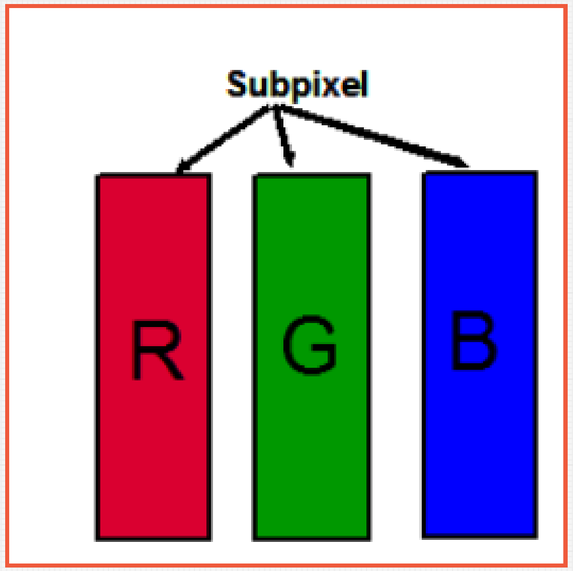
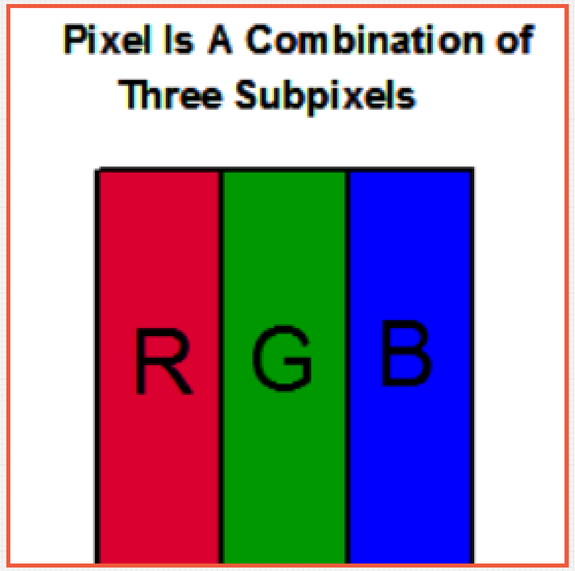


Image
Predicted
Noise Image

Here's the Problem



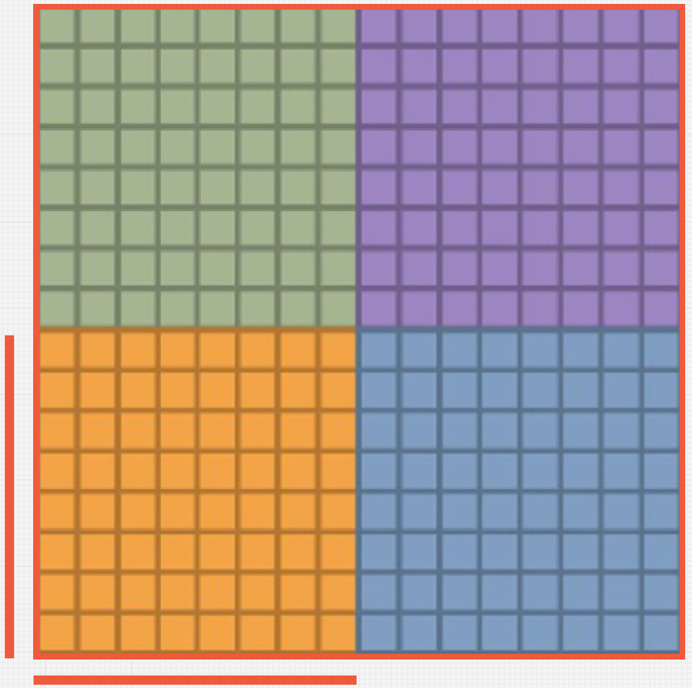


A 512 x 512 image has 786,432 pixels
262,144 for Red
262,144 for Green
262,144 for Blue

Spaces

Pixel Space

latent Space



u

u

x 1/u

E

D

x U

1

1

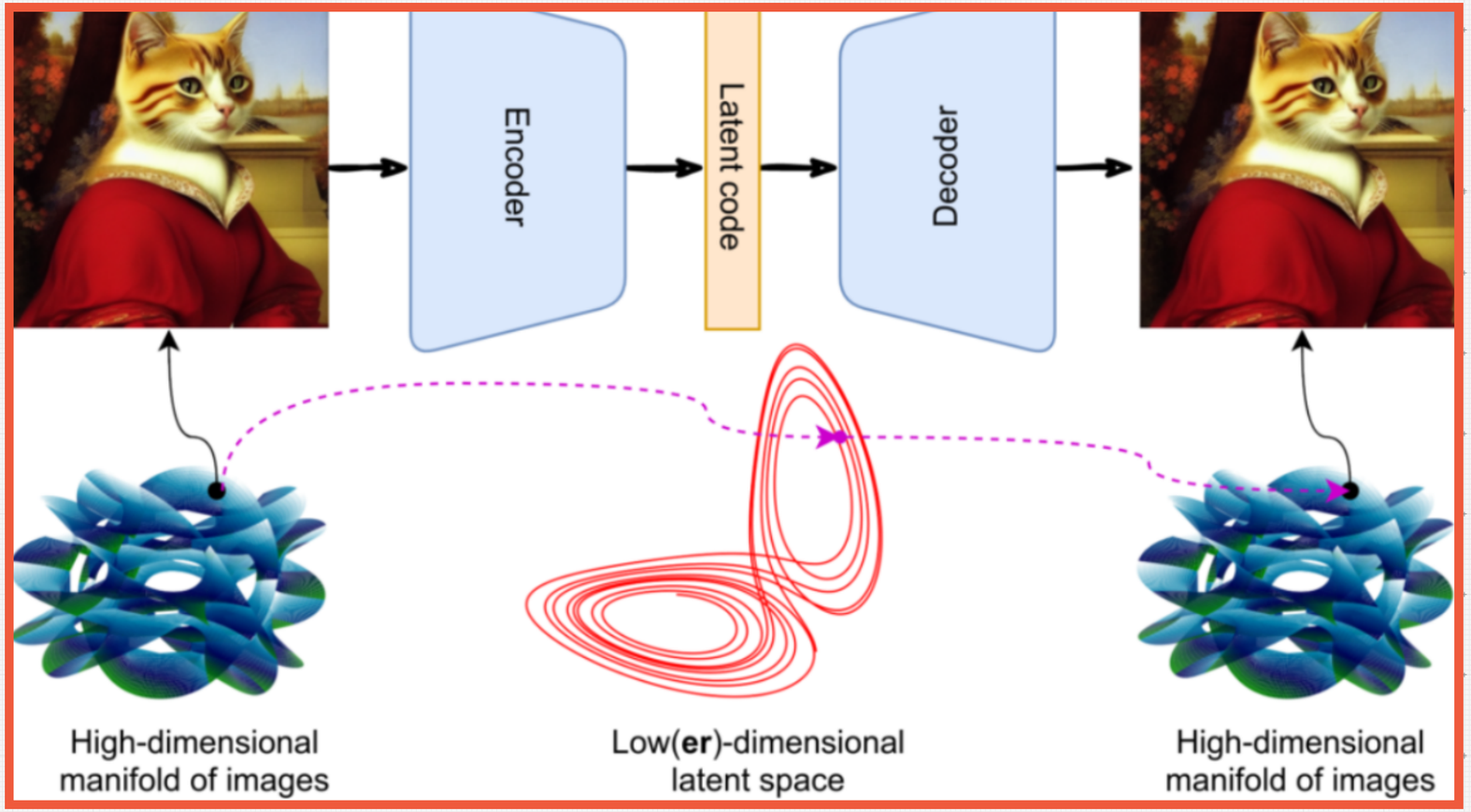
X

Z

How the Speed problem is solved

- The noise image is downscaled into the latent space
- It becomes 4 x 64 x 64 = 16,384
- Noise predictor subtracts noise
- The image is then upscaled into pixel space

Variational AutoEncoders (VAE)



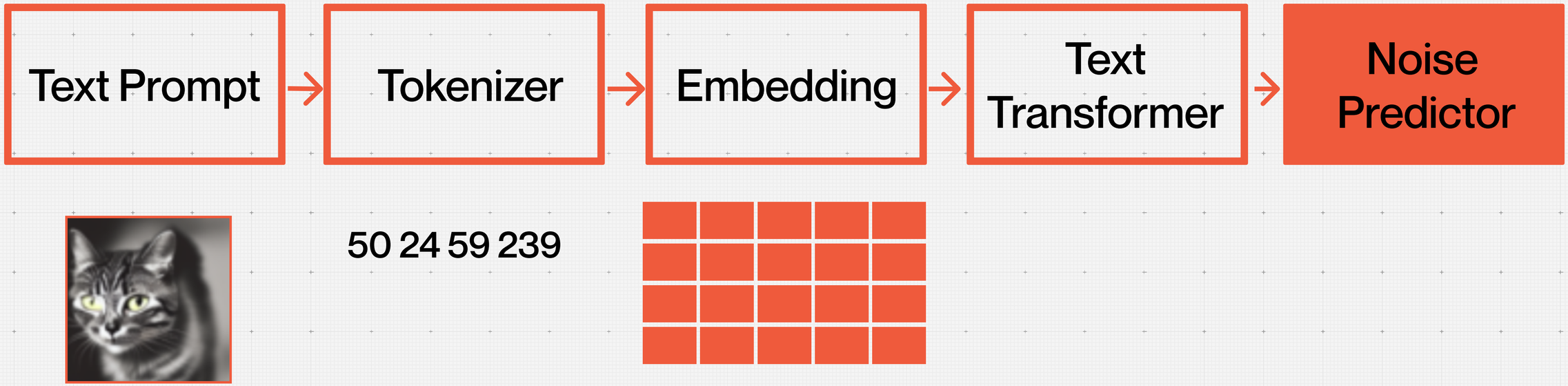
Apply the same training logic in Latent Space

- VAE decodes image into latent space
- Adds latent noise (tensor)
- VAE encodes image back to pixel space

This is why it takes longer VRAM and GPU power to generate images of higher resolution. Thus Upscalers are required.

Where does the text come in?

Conditioning drives the noise towards the desired results. It uses text as a sense of direction.



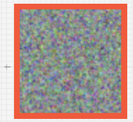
Tokenizers

- Converts words into tokens. Stable Diffusion uses CLIP’s Tokenizer. Every token is associated with a number.
- “Dream” and “Shape” are 2 tokens. “Dreamshape” is 2 tokens as well.

Embedding & Text Transformer

- It’s a 768-value vector. Every token has a unique embedding vector.
- Some words are closely related like woman, lady and girl.
- The text-embedded vector goes into a text-transformer. The transformer can also take in depth-maps, images and class labels as inputs. It then processes the data.

Gaussian Noise
1x4x64x64



Prompt - A dog wearing a hat

CLIP Model

Tokenizer

Token to embedding

Latents
1x4x64x64

Text Embeddings
1x77x768

U-Net

Schedular algorithm ro add noise

Conditions latents
1x4x64x64

VAE

Output Image
(3x512x512)

Some Resources

- https://jalammar.github.io/illustrated-stable-diffusion/
- https://stable-diffusion-art.com/how-stable-diffusion-work/
- https://colab.research.google.com/github/sagiodev/stablediffusion_webui/blob/master/Stable_Diffusion_tokenizer_and_embedding_SDA.ipynb
- https://projector.tensorflow.org/
- https://arxiv.org/abs/2112.10752