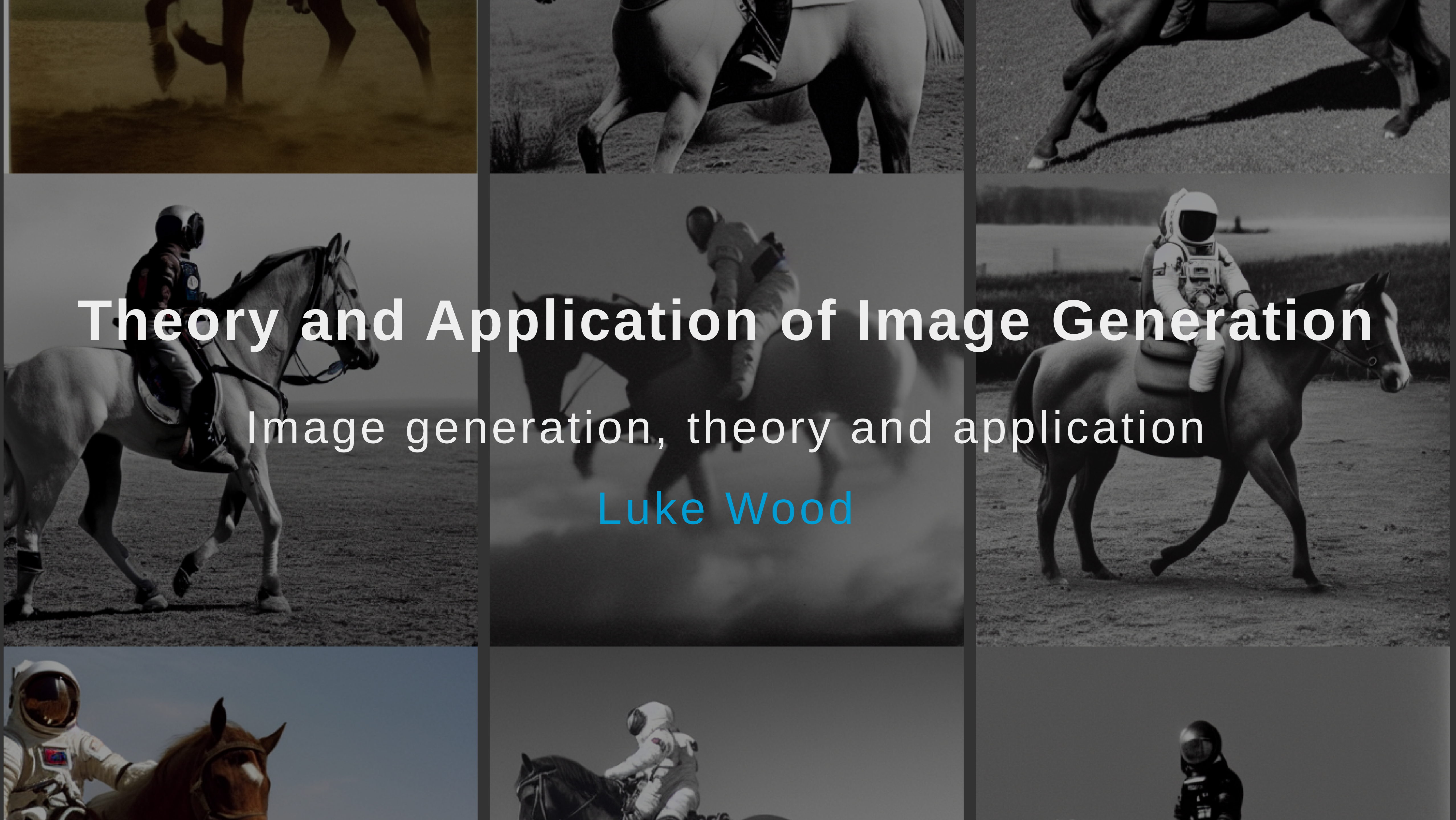


Theory and Application of Image Generation

Image generation, theory and application

Luke Wood

The Code, Slides, Demos

- Slides (PDF): https://lukewood.github.io/devoxx/index.pdf
- Slides (Web): https://lukewood.github.io/devoxx
- Code (for the slides): https://github.com/LukeWood/devoxx

About me

- From San Diego
- Work on the Keras team
- Last year~ on KerasCV
- Pursuing Doctorate at UC San Diego

Background in Generative Modeling

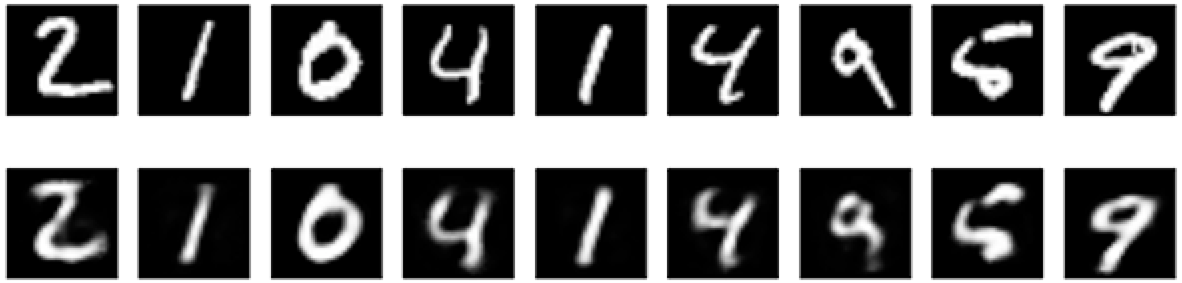
- ML since 2015
- Generative modeling since 2016 (off & on)
- Recent work on StableDiffusion in KerasCV

Generative modeling, why should you care...

Historically you could....

Generate fake shoe pictures

Learn the latent space of a dataset!



(More on this later...)

Generate DeepFakes

All quite interesting...

- but nothing particularly useful
- too difficult to control

Until... DALL-E 2!

TEXT DESCRIPTION

An astronaut   Teddy bears   A bowl of soup

riding a horse   lounging in a tropical resort in space   playing basketball with cats in space

in a photorealistic style   in the style of Andy Warhol   as a pencil drawing

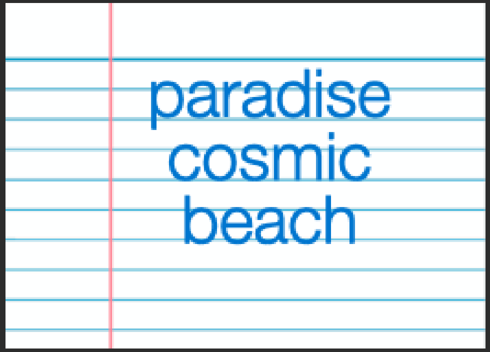
→

DALL·E 2

And then... StableDiffusion!

Stable Diffusion is a deep learning, text-to-image model released by startup StabilityAI in 2022.

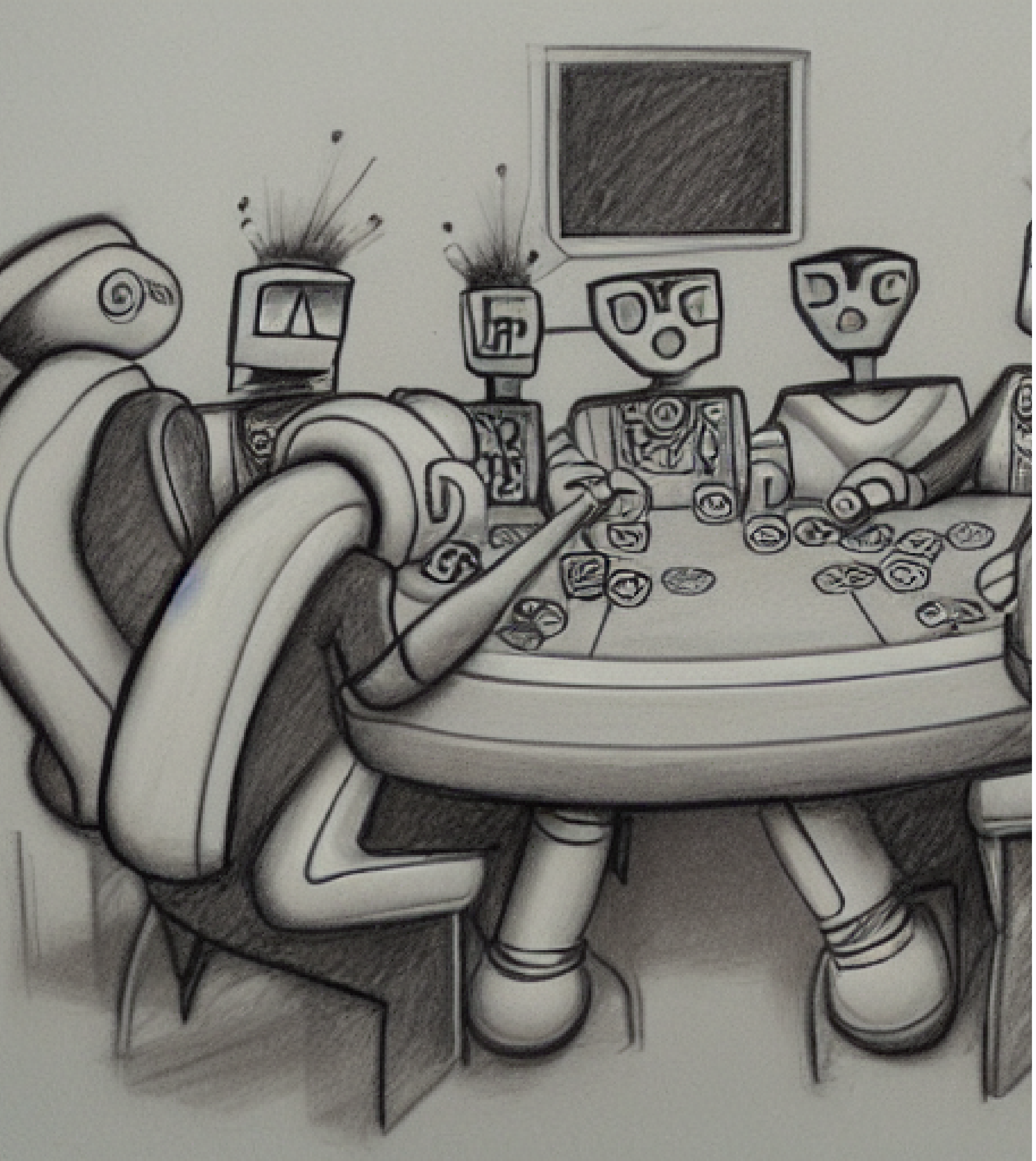
Most importantly, StableDiffusion is 100% open source... and generously licensed


paradise
cosmic
beach


STABLE
DIFFUSION

"A gentleman otter in a 19th century portrait"

"A cute magical flying dog, fantasy art drawn by Disney concept artists"



"pencil sketch of robots playing poker"

"Multicolor hyperspace"

But that's not all!

Image to image workflows GUIDED by text


Pirate ship
STABLE DIFFUSION

Image to image inpainting (as seen in the intro)!

… and outpainting!

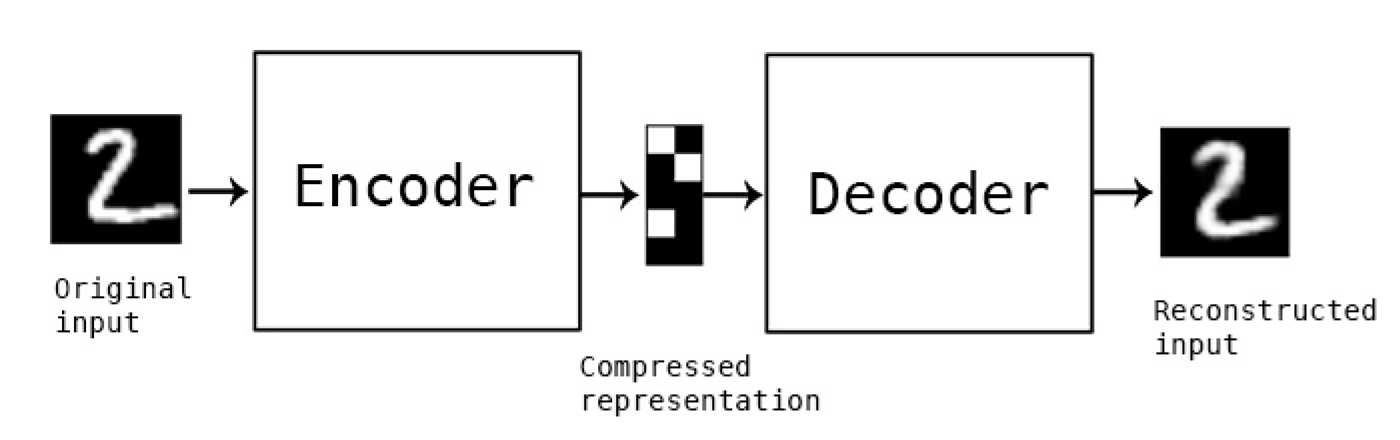
… and variation generation!

Now that I have your attention...

Lets take a step back! How does this all work?

Representations & Continuity

AutoEncoders



- AutoEncoders: travel back to 1987
- early days of ML
- no large scale data
- unfortunately, no good visual results for you!
- backprop "without a teacher"

Flash forward to the 2010s

TensorFlow, GPUs, large datasets

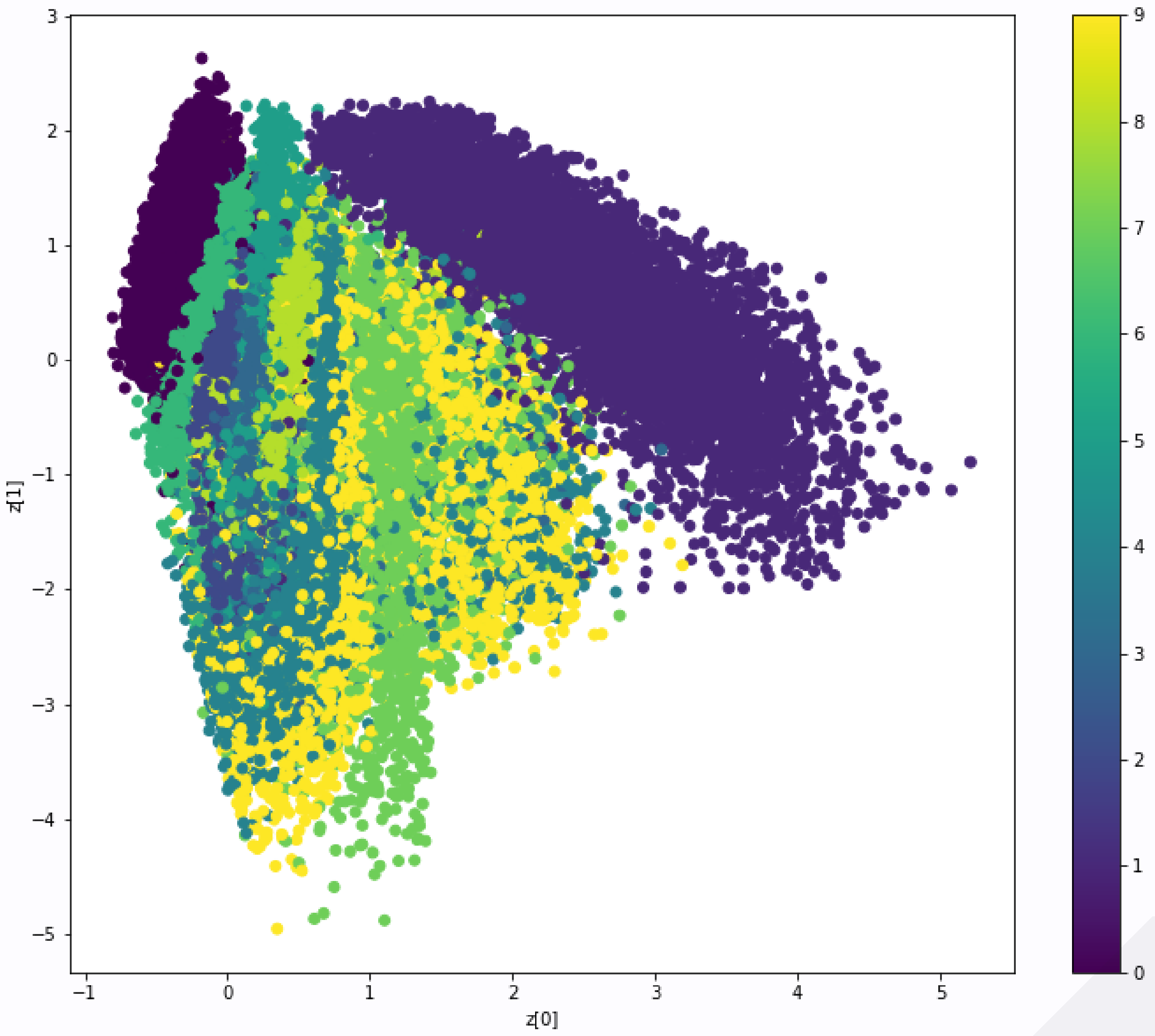
AutoEncoders are a form of compression

Caveats

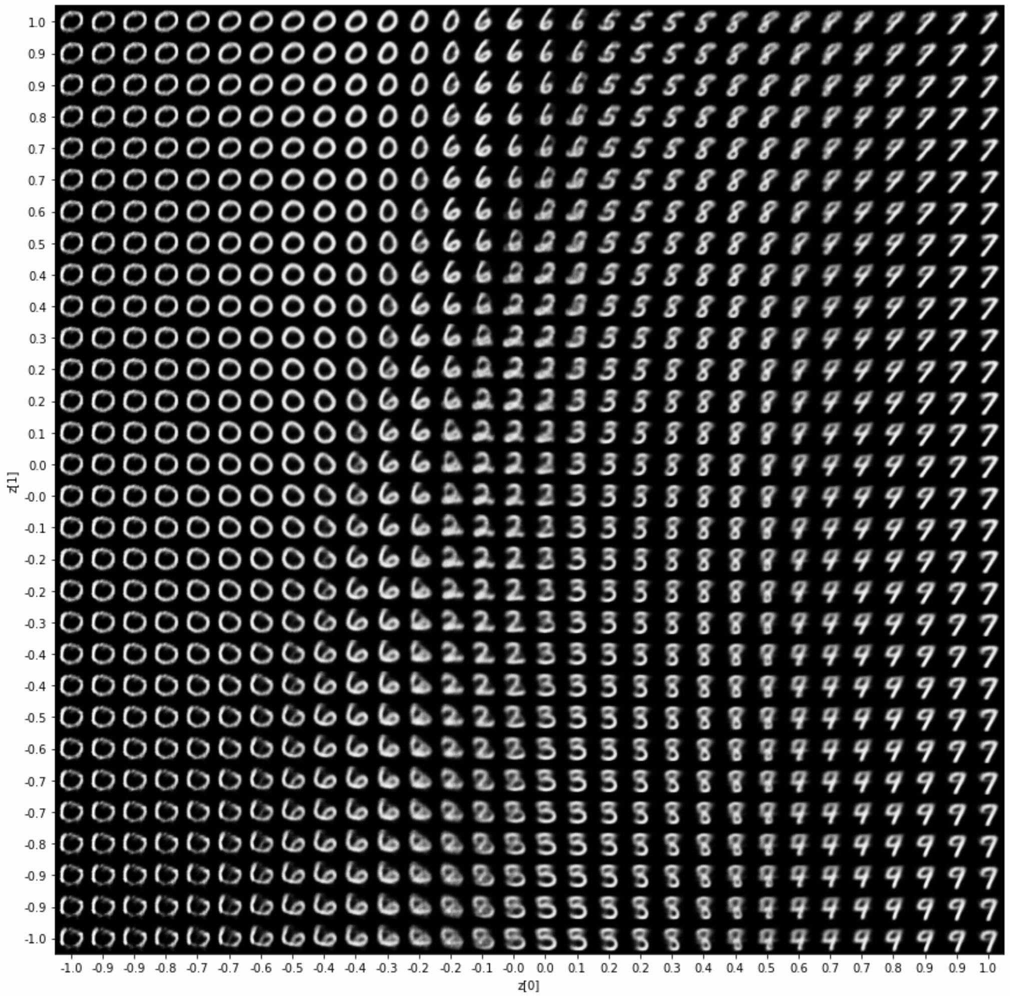
- data specific
- lossy
- "They are rarely used in practical applications" - Keras blog in 2016

... but what happens in between real samples?

```
def plot_label_clusters(vae, data, labels):
    # display a 2D plot of the digit classes in the latent space
    z_mean, _, _ = vae.encoder.predict(data)
    plt.figure(figsize=(12, 10))
    plt.scatter(z_mean[:, 0], z_mean[:, 1], c=labels)
    plt.colorbar()
    plt.xlabel("z[0]")
    plt.ylabel("z[1]")
    plt.show()


(x_train, y_train), _ = keras.datasets.mnist.load_data()
x_train = np.expand_dims(x_train, -1).astype("float32") / 255

plot_label_clusters(vae, x_train, y_train)
```



Generate new images!

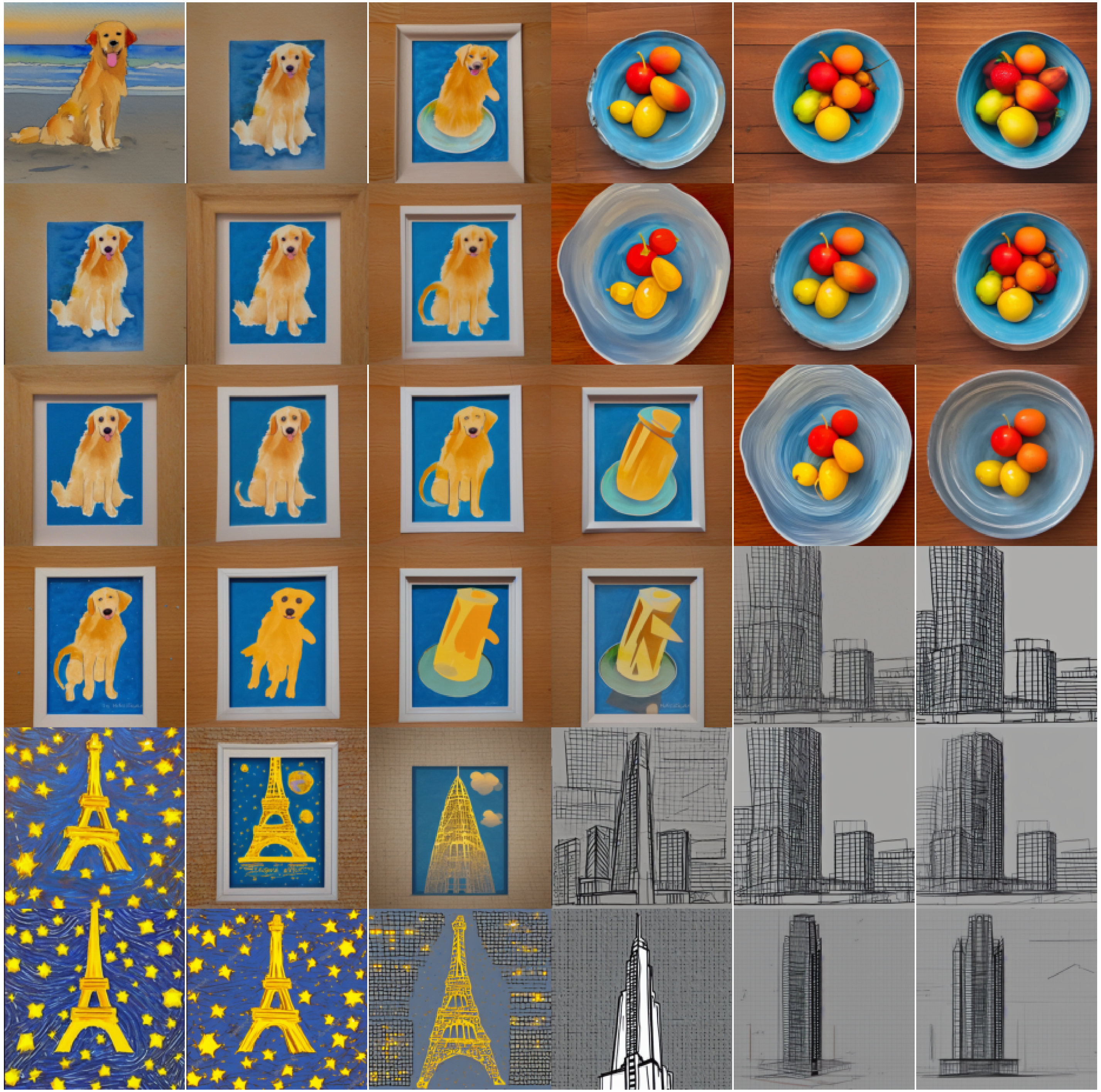


Continuity!

"Latent space walking, or latent space exploration, is the process of sampling a point in latent space and incrementally changing the latent representation. Its most common application is generating animations where each sampled point is fed to the decoder and is stored as a frame in the final animation. For high-quality latent representations, this produces coherent-looking animations. These animations can provide insight into the feature map of the latent space, and can ultimately lead to improvements in the training process."

Panda ➡ Plane

Dog ➡ Bowl of fruit



A quick aside on Variational AutoEncoders (VAEs)...

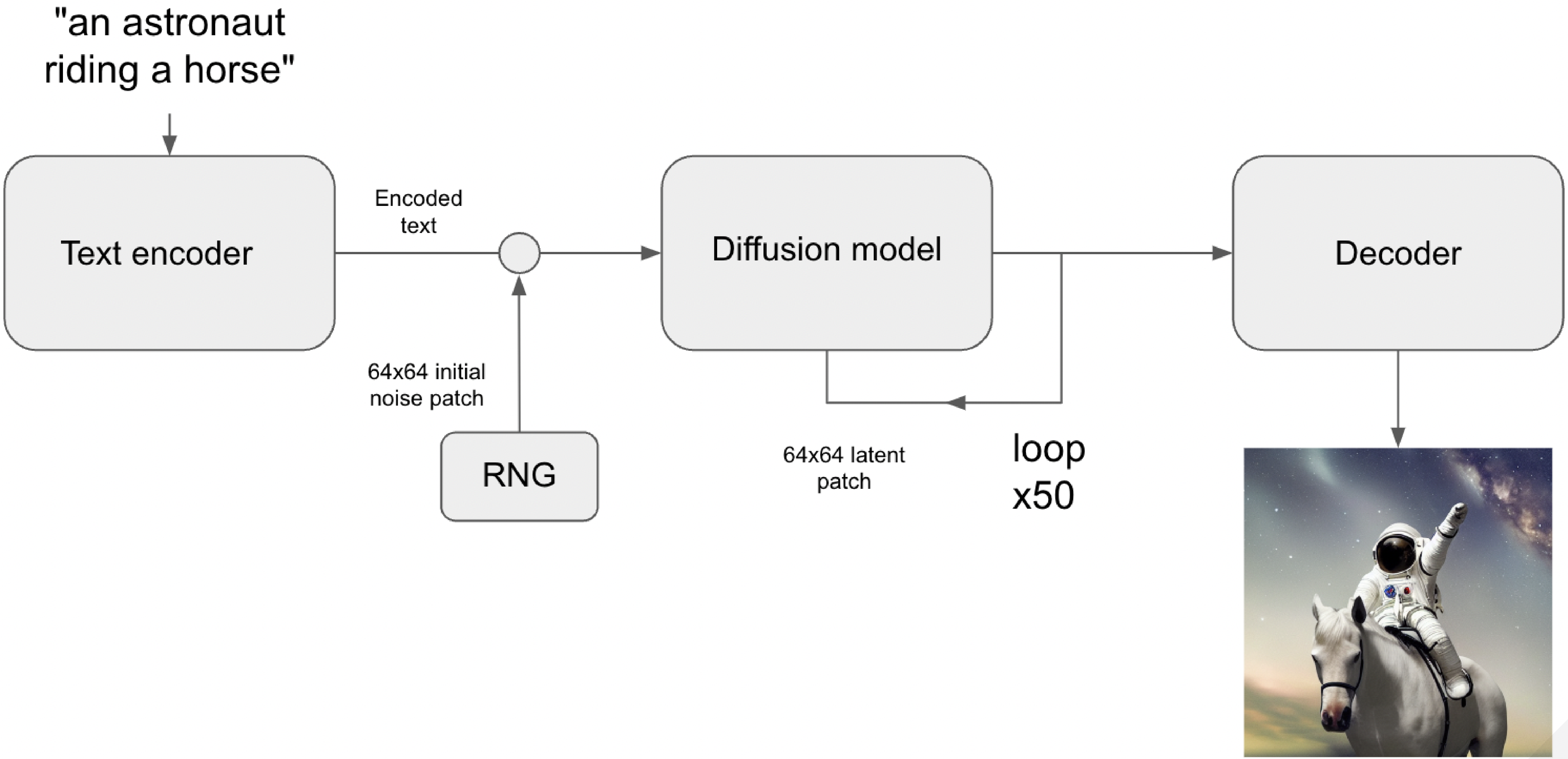
```
class Sampling(layers.Layer):
    """Uses (z_mean, z_log_var) to sample z, the vector encoding a digit."""

    def call(self, inputs):
        z_mean, z_log_var = inputs
        batch = tf.shape(z_mean)[0]
        dim = tf.shape(z_mean)[1]
        epsilon = tf.keras.backend.random_normal(shape=(batch, dim))
        return z_mean + tf.exp(0.5 * z_log_var) * epsilon
```

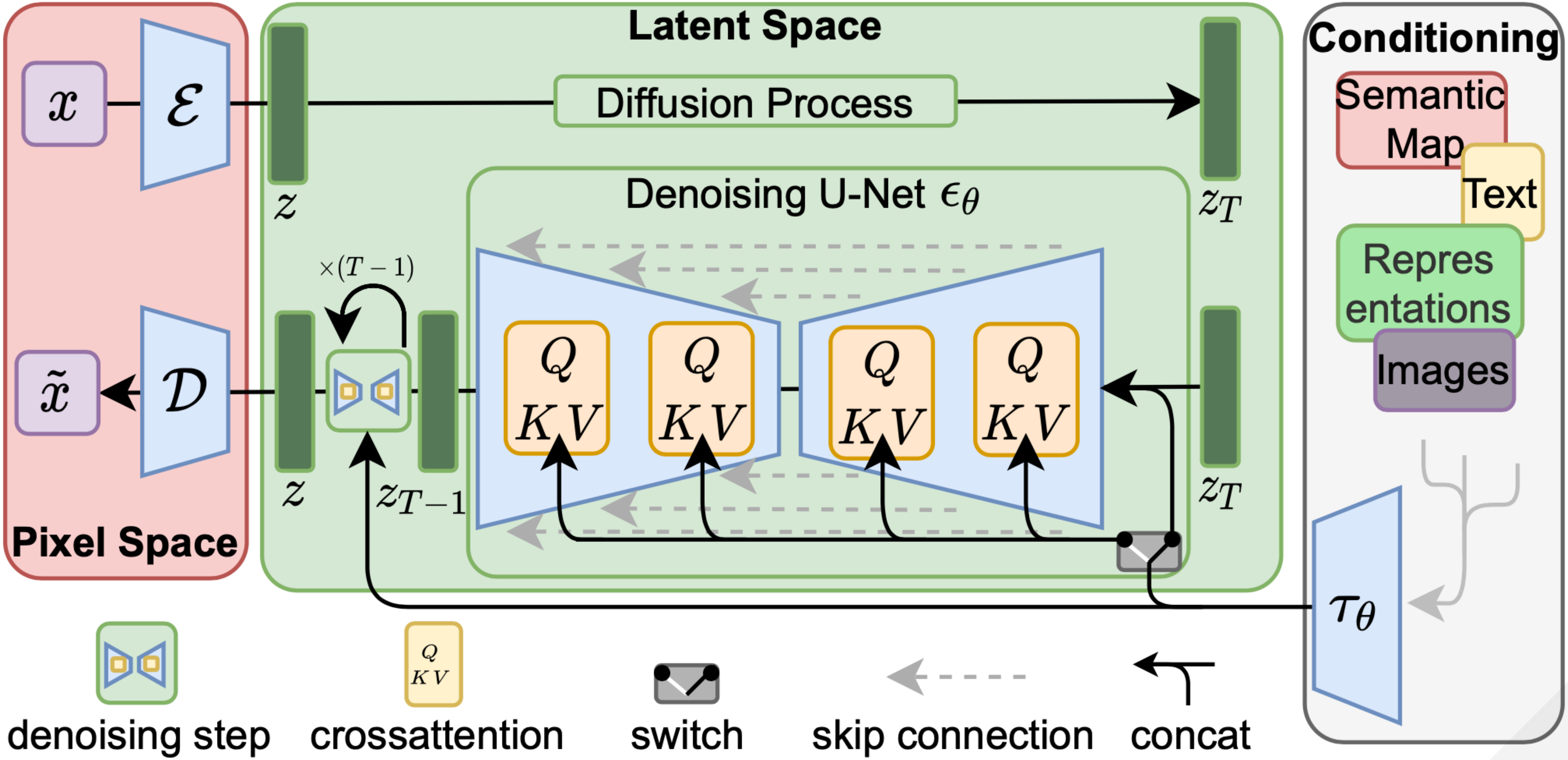
Any Questions?

(on continuity only please)

Congratulations!

You now understand approximately 1/4 of StableDiffusion.

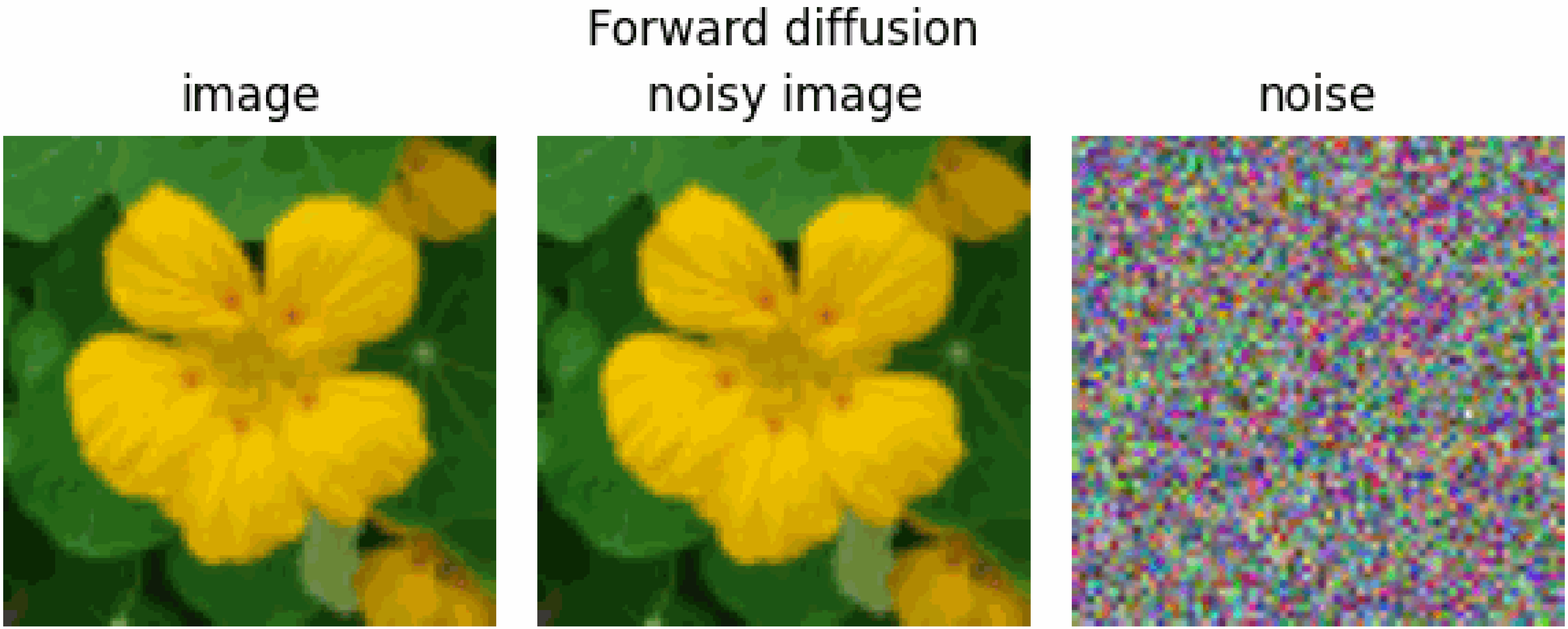

"an astronaut riding a horse"
Text encoder
Encoded text
64x64 initial noise patch
RNG
Diffusion model
64x64 latent patch
loop x50
Decoder

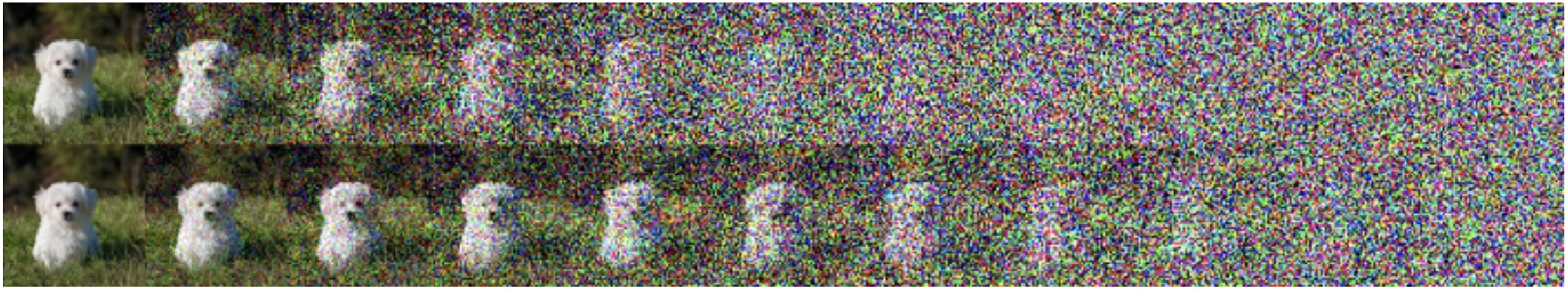


Latent Space
Pixel Space
Conditioning
x
$\mathcal{E}$
z
Diffusion Process
z_T
Denoising U-Net ϵ_θ
$\times(T-1)$
$\tilde{x}$
$\mathcal{D}$
z
z_{T-1}
z_T
Q
KV
Semantic Map
Text
Repres entations
Images
τ_θ
denoising step
crossattention
switch
skip connection
concat

Diffusion Models

Denoising Diffusion Probabilistic Models, 2020


Forward diffusion
image
noisy image
noise



Super-resolution

- Image Super-Resolution using an Efficient Sub-Pixel CNN
- Enhanced Deep Residual Networks for single-image super-resolution

Push super resolution to the limit!

- start from pure noise
- proposed in 2020

More reading on keras.io

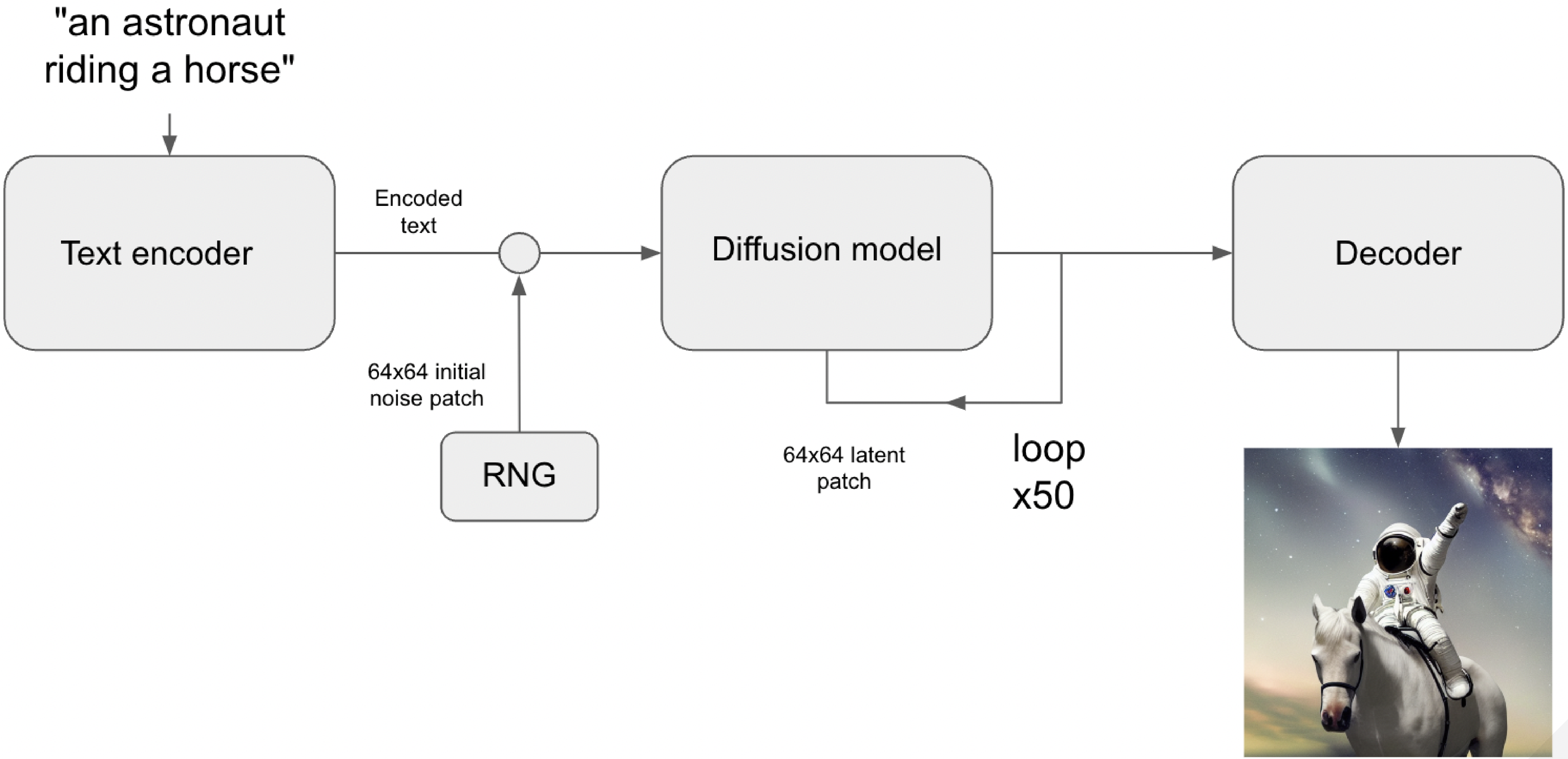
- Denoising Diffusion Implicit Models

Any questions?

(On diffusion models)

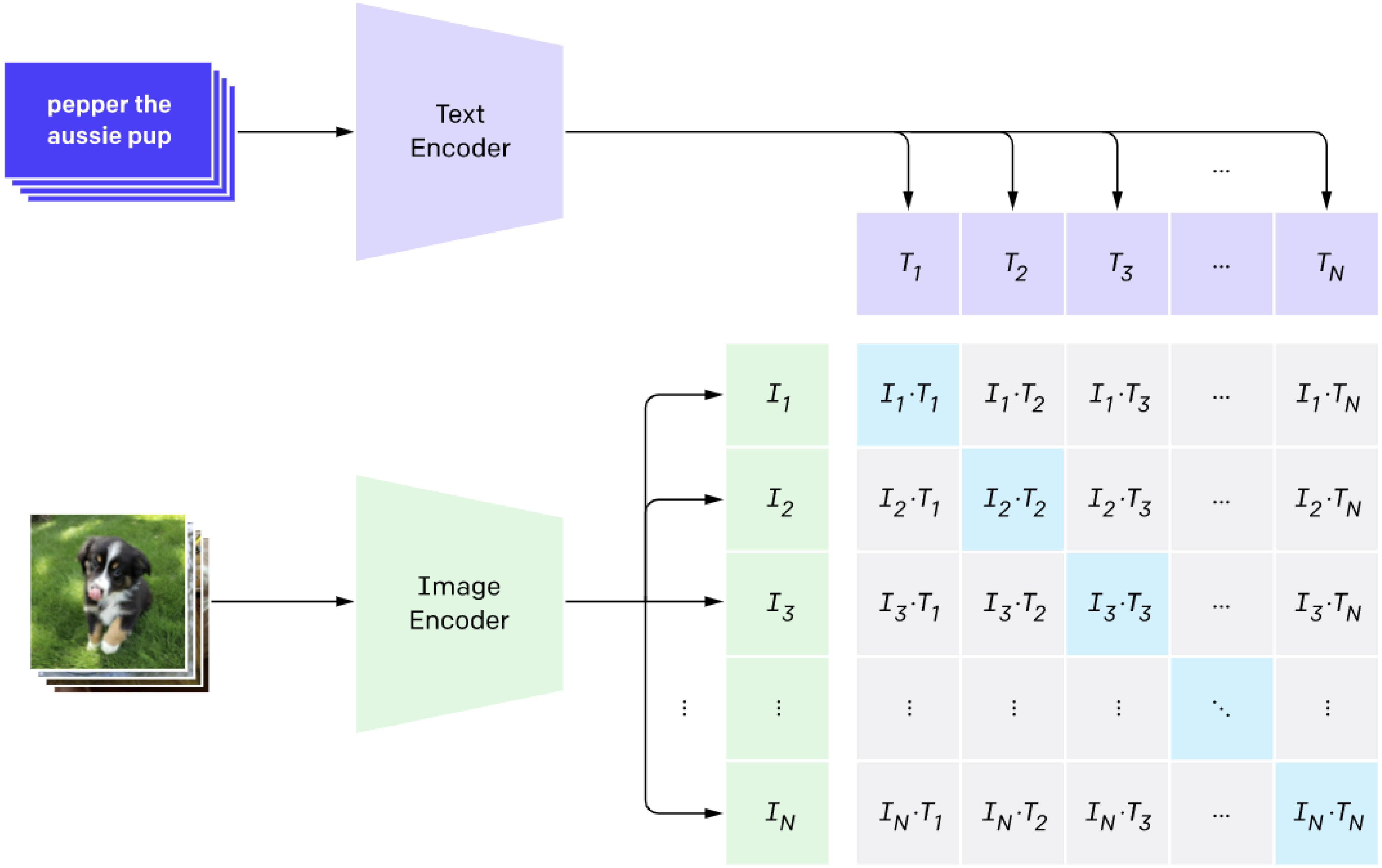
Latent diffusion models

- improves efficiency
- use VAE decoder
- UNet

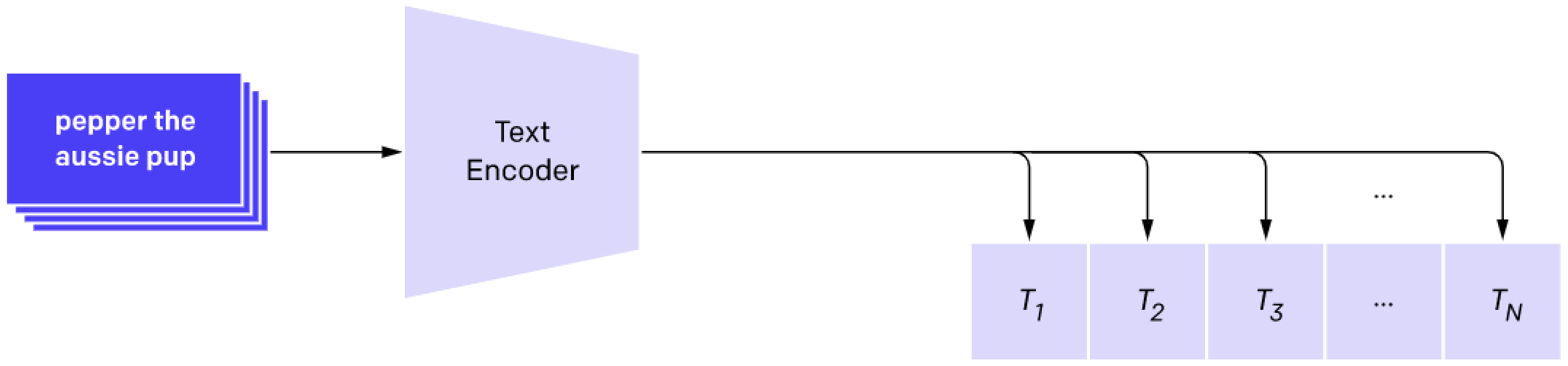

"an astronaut riding a horse"
Text encoder
Encoded text
64x64 initial noise patch
RNG
Diffusion model
64x64 latent patch
loop x50
Decoder

CLIP

... what you need to know

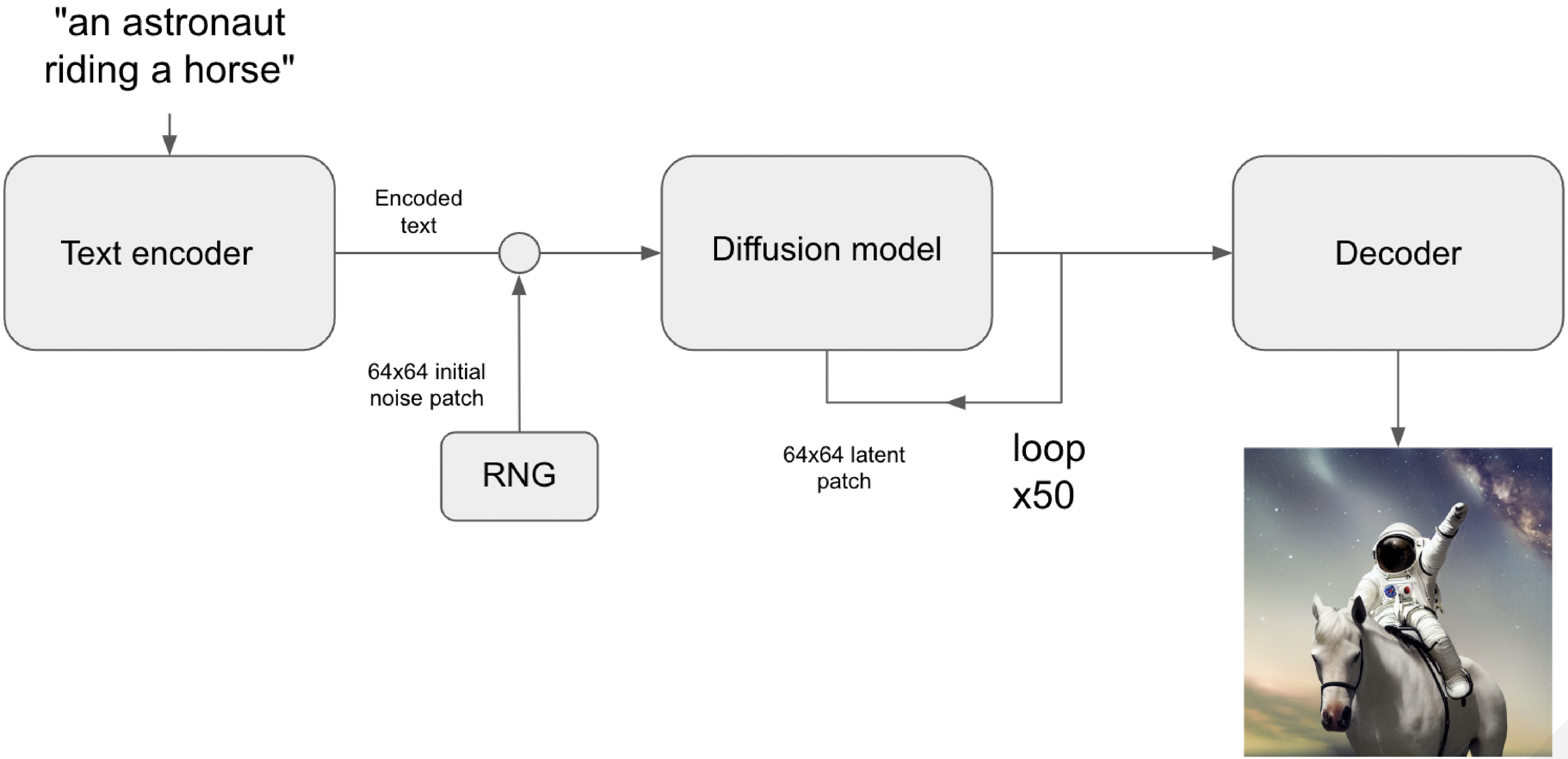

pepper the aussie pup
Text Encoder
Image Encoder
T_1 T_2 T_3 ... T_N
I_1 I_2 I_3 ⋮ I_N
$I_1 \cdot T_1$ $I_1 \cdot T_2$ $I_1 \cdot T_3$ ... $I_1 \cdot T_N$
$I_2 \cdot T_1$ $I_2 \cdot T_2$ $I_2 \cdot T_3$ ... $I_2 \cdot T_N$
$I_3 \cdot T_1$ $I_3 \cdot T_2$ $I_3 \cdot T_3$ ... $I_3 \cdot T_N$
$I_N \cdot T_1$ $I_N \cdot T_2$ $I_N \cdot T_3$ ... $I_N \cdot T_N$

We just need the text encoder



CLIP

More reading available on the OpenAI blog post

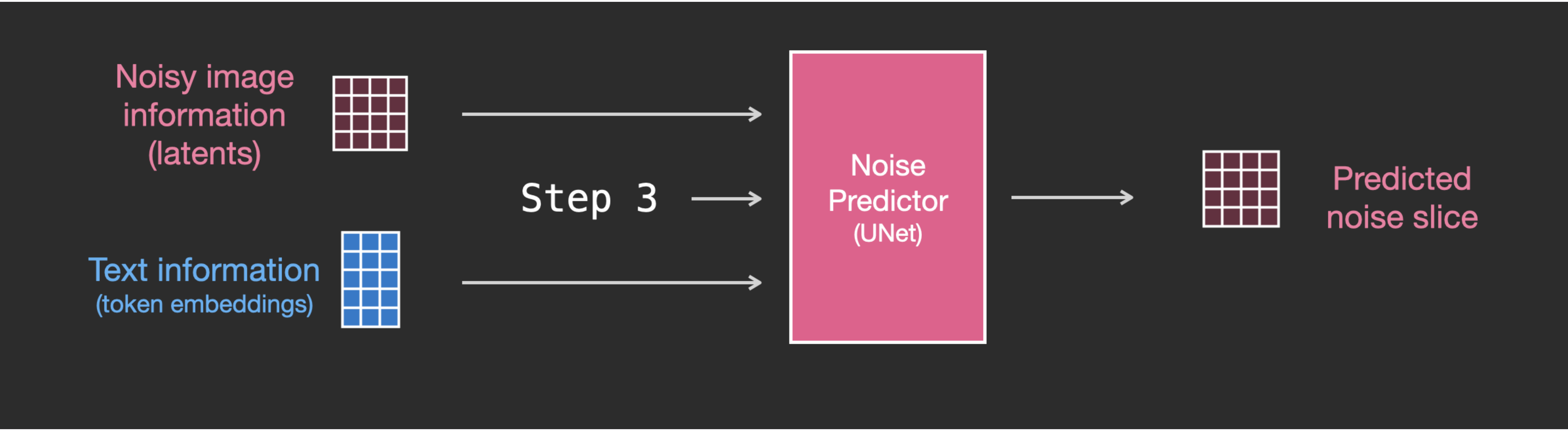

"an astronaut riding a horse"
Text encoder
Encoded text
64x64 initial noise patch
RNG
Diffusion model
64x64 latent patch
loop x50
Decoder

the Final Piece…

Conditioning!

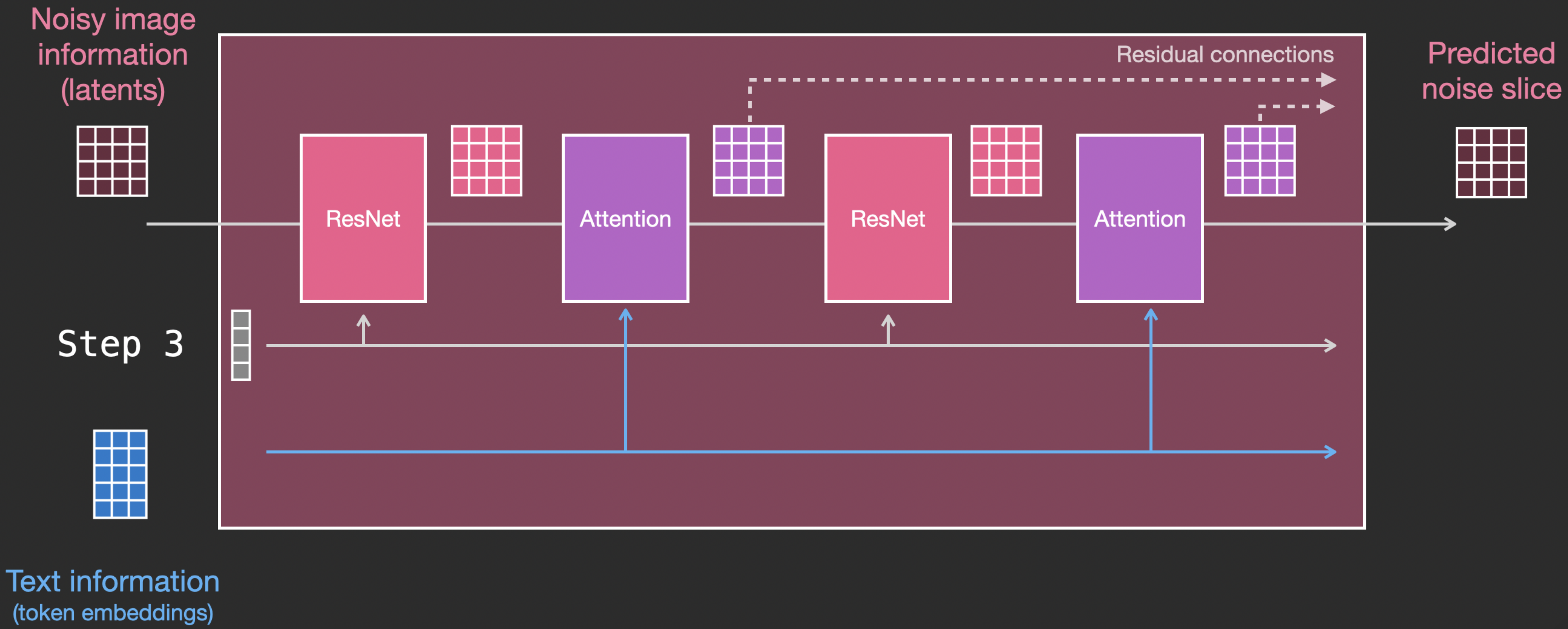
Conditioning

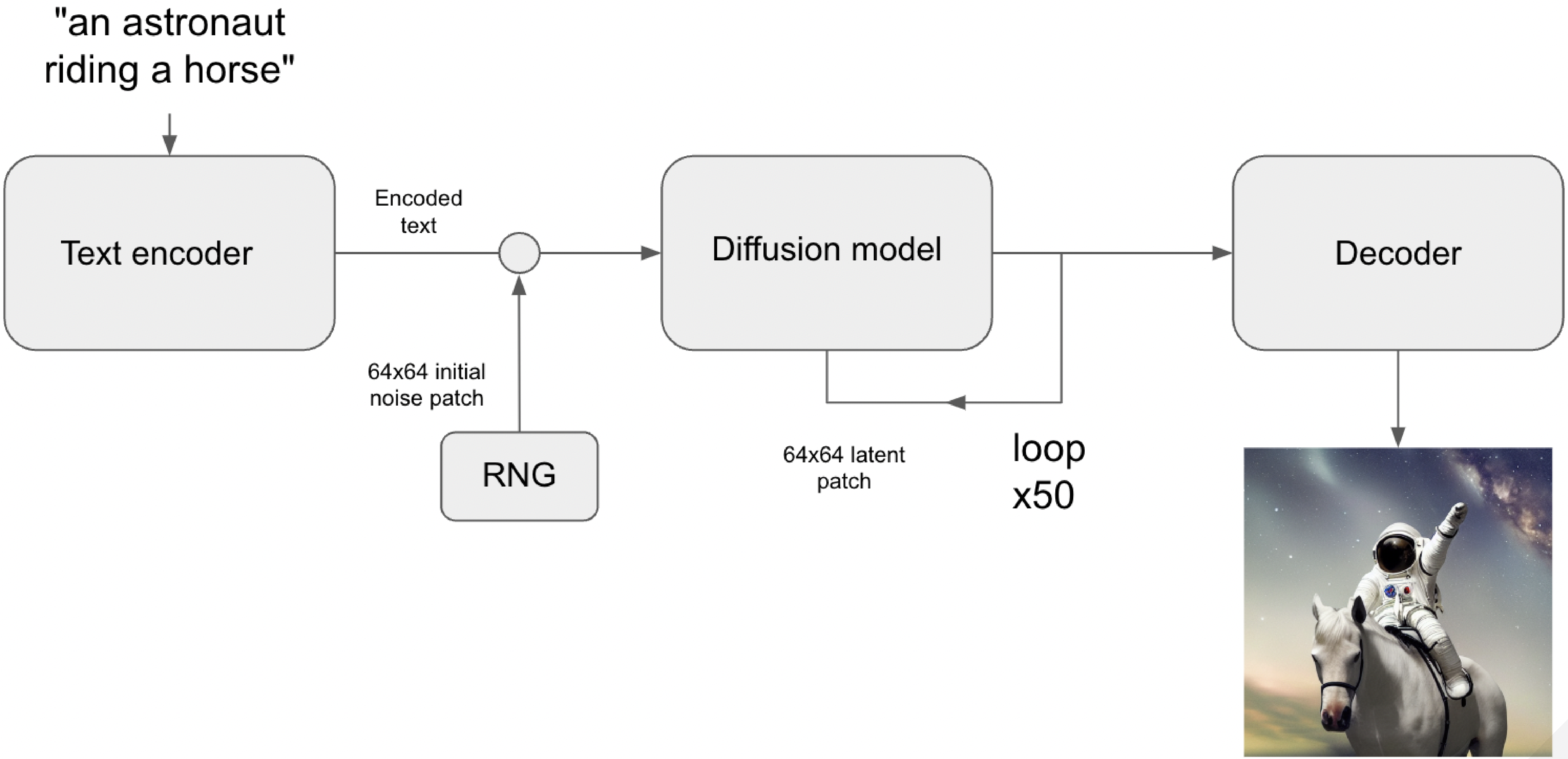
- classic deep learning
- concatenate
- 64x64x3 ➡️ 64x64x4

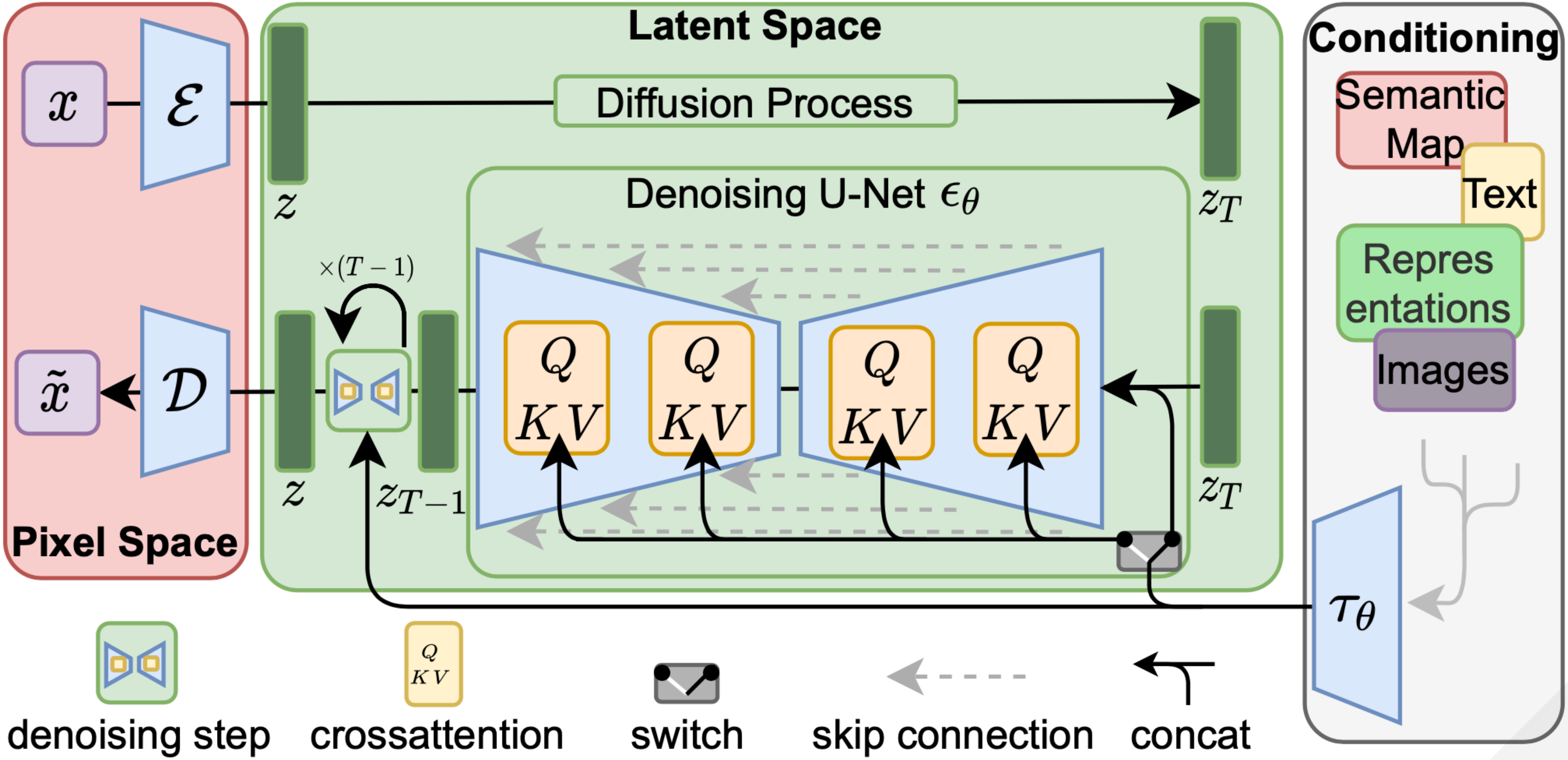


Noisy image information (latents)
Text information (token embeddings)
Step 3
Noise Predictor (UNet)
Predicted noise slice

Noise Predictor with Text Conditioning (UNet with attention)




"an astronaut riding a horse"
Text encoder
Encoded text
64x64 initial noise patch
RNG
Diffusion model
64x64 latent patch
loop x50
Decoder



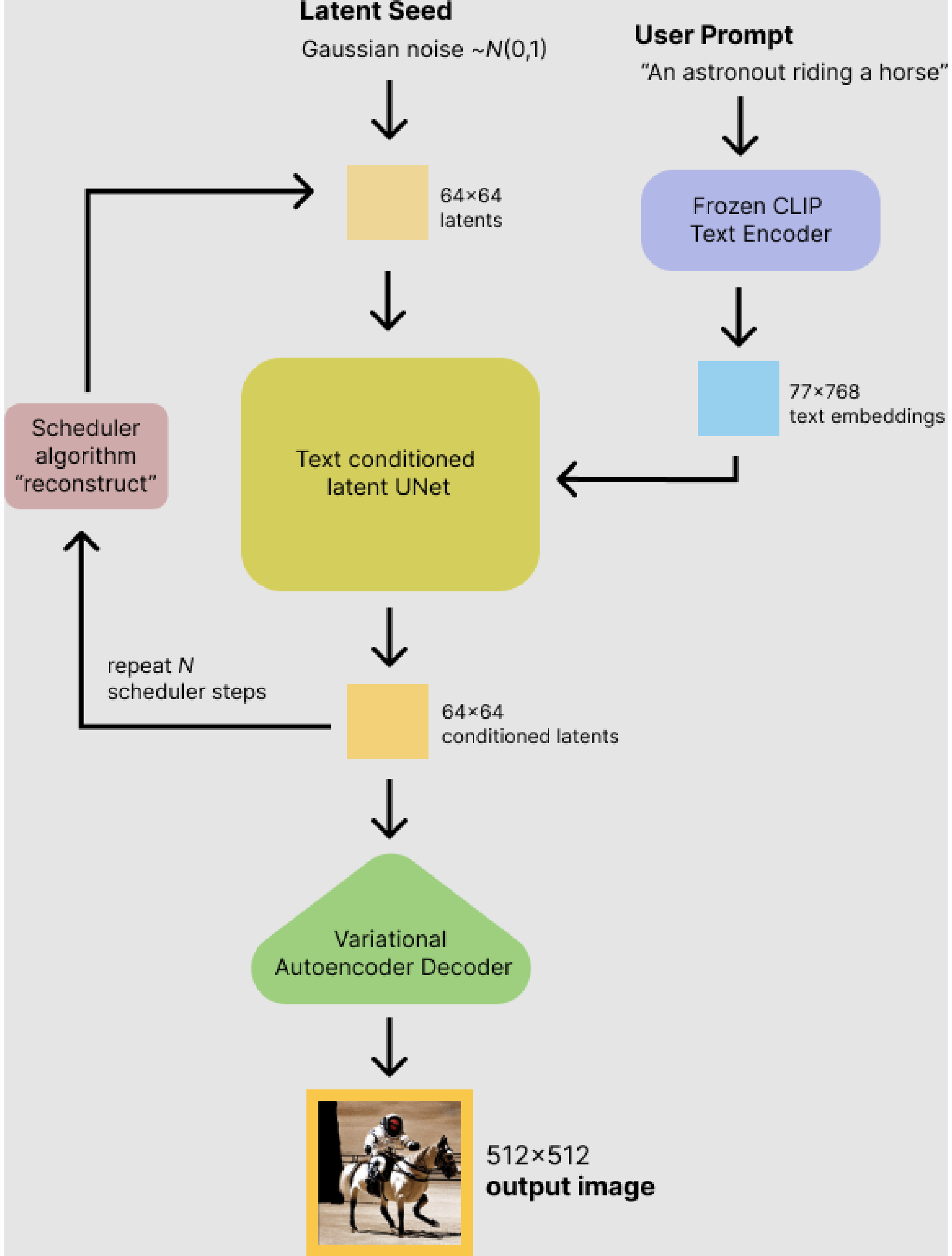
Latent Space
Conditioning
Diffusion Process
Semantic Map
Text
Repres entations
Images
Denoising U-Net ϵ_θ
Pixel Space
x
$\mathcal{E}$
z
z_T
$\times(T-1)$
$\tilde{x}$
$\mathcal{D}$
z_{T-1}
Q K V
τ_θ
denoising step
crossattention
switch
skip connection
concat

That's All!

You now know how StableDiffusion works!

How do I use it?

Text to Image Generation


Latent Seed
Gaussian noise ~N(0,1)
User Prompt
"An astronout riding a horse"
64×64
latents
Frozen CLIP
Text Encoder
77×768
text embeddings
Scheduler
algorithm
"reconstruct"
Text conditioned
latent UNet
repeat N
scheduler steps
64×64
conditioned latents
Variational
Autoencoder Decoder
512×512
output image

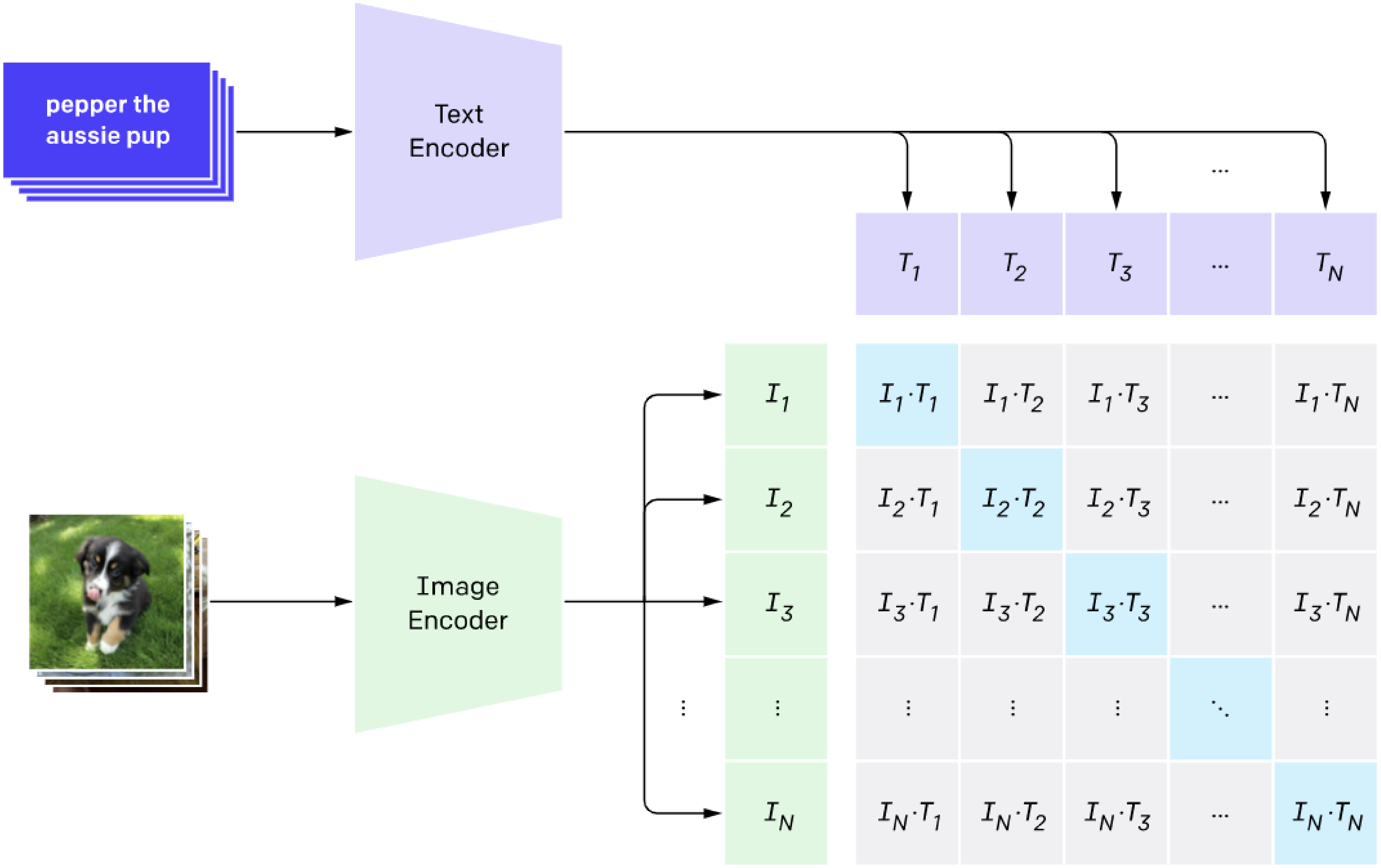
"An astronaut riding a horse"

Code:

```python
from tensorflow import keras
import keras_cv

keras.mixed_precision.set_global_policy("mixed_float16")
model = keras_cv.models.StableDiffusion(jit_compile=True)

images = model.text_to_image(
    "Teddy bears conducting machine learning research",
    batch_size=4,
)
plot_images(images)
```

Variation generation

Remember CLIP?


pepper the aussie pup
Text Encoder
Image Encoder
T_1 T_2 T_3 ... T_N
I_1 I_2 I_3 ⋮ I_N
$I_1 \cdot T_1$ $I_1 \cdot T_2$ $I_1 \cdot T_3$ ... $I_1 \cdot T_N$
$I_2 \cdot T_1$ $I_2 \cdot T_2$ $I_2 \cdot T_3$ ... $I_2 \cdot T_N$
$I_3 \cdot T_1$ $I_3 \cdot T_2$ $I_3 \cdot T_3$ ... $I_3 \cdot T_N$
$I_N \cdot T_1$ $I_N \cdot T_2$ $I_N \cdot T_3$ ... $I_N \cdot T_N$

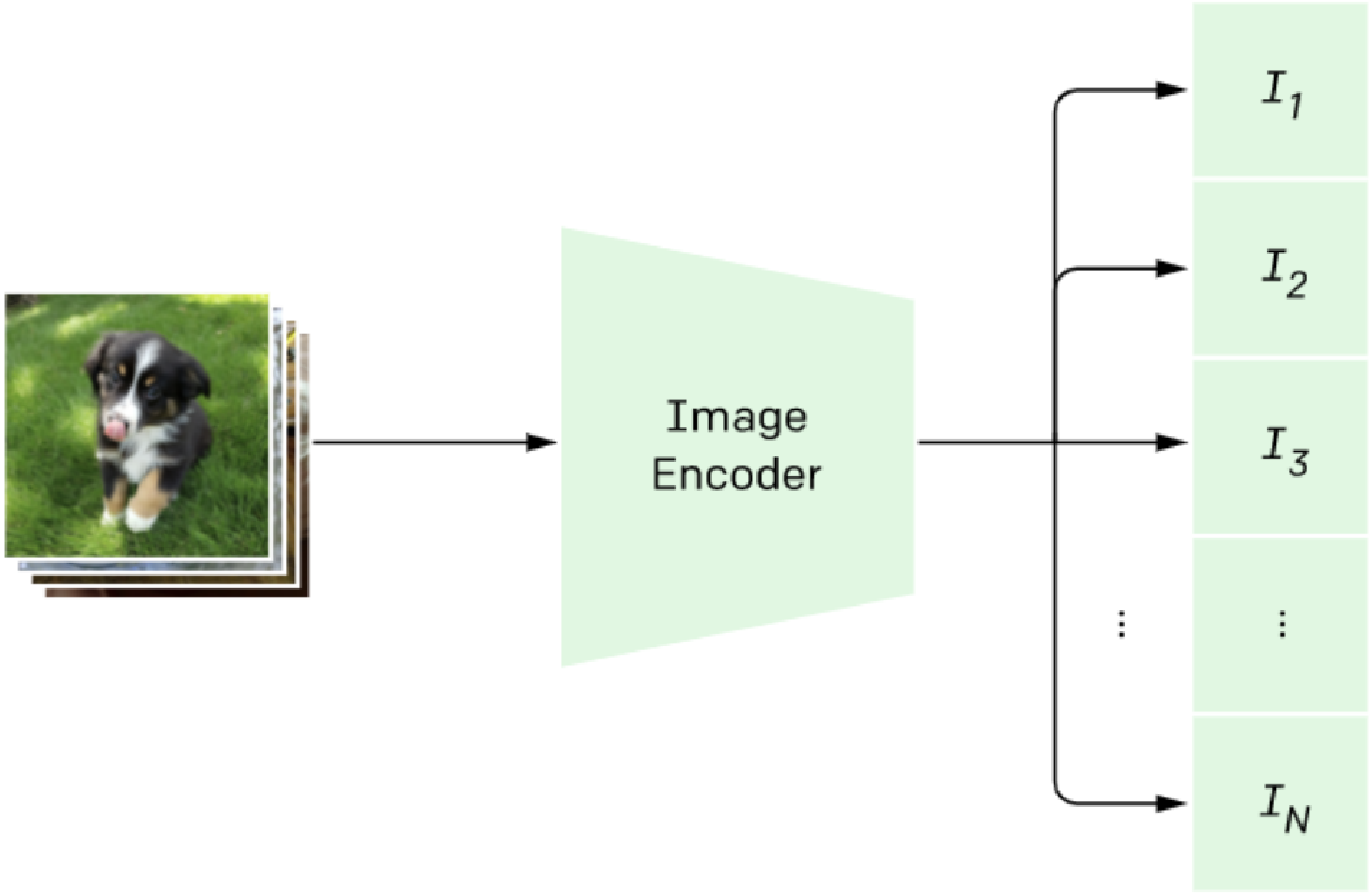


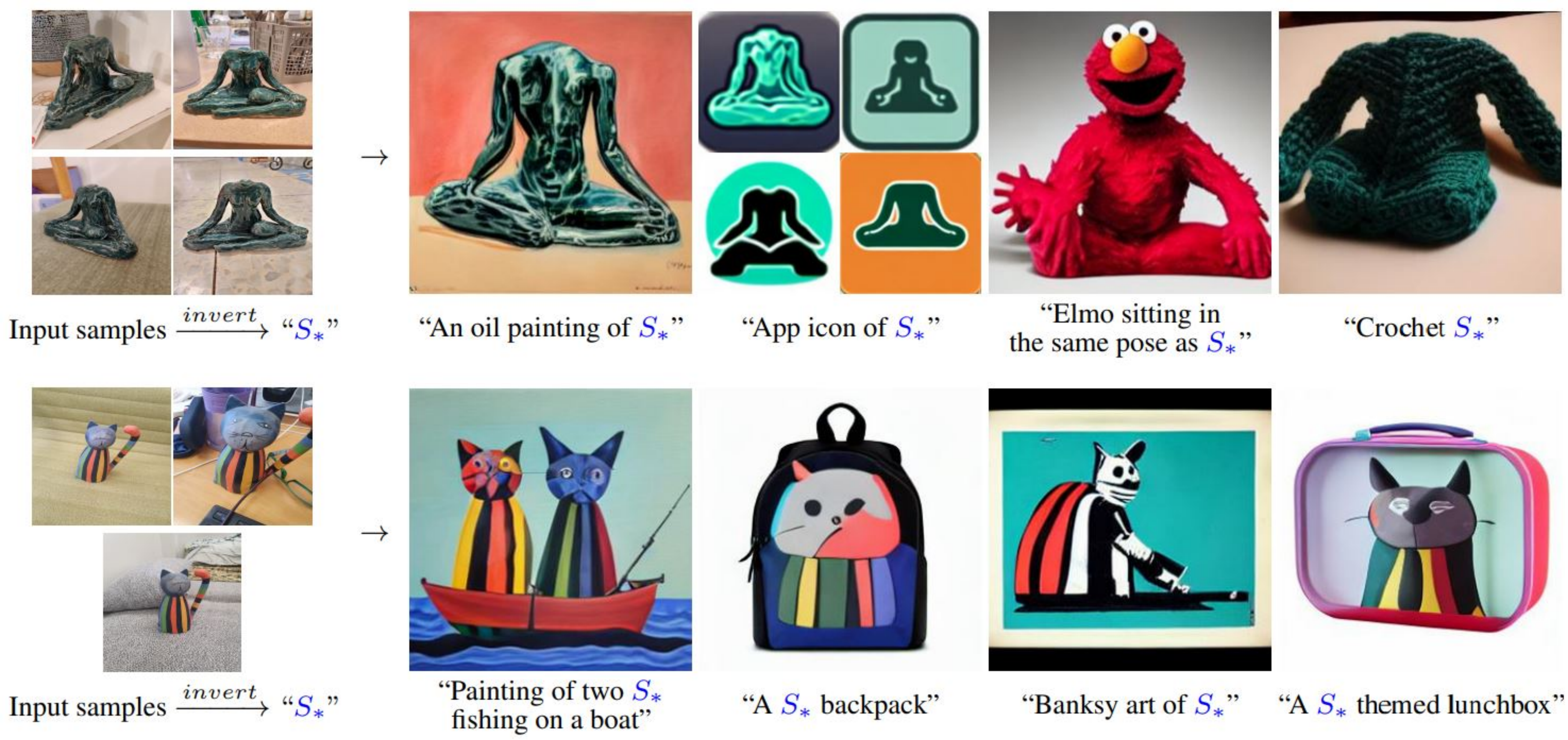
Image
Encoder
I_1
I_2
I_3
I_N

Switch it out!

It's really that easy!

Textual Inversion

Teach new concepts to StableDiffusion!


Input samples $\xrightarrow{invert}$ "S_*"
"An oil painting of S_*"
"App icon of S_*"
"Elmo sitting in the same pose as S_*"
"Crochet S_*"
Input samples $\xrightarrow{invert}$ "S_*"
"Painting of two S_* fishing on a boat"
"A S_* backpack"
"Banksy art of S_*"
"A S_* themed lunchbox"

Step 1: collect 3-5 images of your object

```python
urls = [
    "https://i.imgur.com/VIedH1X.jpg",
    "https://i.imgur.com/iLkM4Ar.jpg",
    "https://i.imgur.com/eBw13hE.png",
]
files = [tf.keras.utils.get_file(origin=url) for url in urls]
# Resize images
resize = keras.layers.Resizing(height=512, width=512, crop_to_aspect_ratio=True)
images = [keras.utils.load_img(img) for img in files]
images = [keras.utils.img_to_array(img) for img in images]
images = np.array([resize(img) for img in images])
visualization.plot_gallery(images, value_range=(0, 255), rows=1, cols=3)
```

Step 2: add a special token to the model vocabulary

```
your_token = '<any-special-name>'
tokenizer.add_token(your_token)
```

Step 3: construct an image-caption dataset

```
your_token = '<any-special-name>'
templates = [
    "a photo of a {}",
    "a rendering of a {}",
    "a cropped photo of the {}",
    "the photo of a {}",
    # ...
]
templates = [t.format(your_token) for t in templates]

# Construct a TensorFlow dataset of the images + tokens
image_dataset = tf.data.Dataset.from_tensor_slices(images)
text_dataset = tf.data.Dataset.from_tensor_slices(templates)
# ... there is a bit more boilerplate to pre-process the text
train_ds = tf.data.Dataset.zip(
  (image_dataset.shuffle(), text_dataset.shuffle())
)
```

Step 4: Fine Tune the TextEncoder with your new dataset!

```
stable_diffusion.diffusion_model.trainable = False
stable_diffusion.decoder.trainable = False
stable_diffusion.text_encoder.trainable = True

trainer = StableDiffusionFineTuner(stable_diffusion, name="trainer")
optimizer = keras.optimizers.SGD(learning_rate=5e-4)
trainer.compile(optimizer=optimizer, loss="mse")

# trainer trains the StableDiffusion model for you.
trainer.fit(
    train_ds,
    epochs=10,
    steps_per_epoch=200
)
```

Results

```
images = stable_diffusion.text_to_image(
    "a photo of <any-special-name> wearing a top hat",
    batch_size=4,
)
plot_images(images)
```

Results

```
images = stable_diffusion.text_to_image(
    "An app icon of <any-special-name>.",
    batch_size=4,
)
plot_images(images)
```

Demo Time

Prompt requests?

Follow along on Colab!

Conclusions

- limitless possibilities
- the power of multi-modal models
- how fast the field is evolving

More Workflows Coming Soon

Other workflows are coming to KerasCV soon!

Other links

- Textual Inversion with Huggingface
- Image variations with lambda labs

Thank you!

- Slides (Web): https://lukewood.github.io/devoxx
- Slides (PDF): https://lukewood.github.io/devoxx/index.pdf
- Keras
- KerasCV

References:

- The Illustrated Stable Diffusion
- DALL-E: Introducing Outpainting
- keras.io: Variational AutoEncoder
- keras.io: A walk through latent space with Stable Diffusion
- Denoising Diffusion Implicit Models
- CLIP: Connexting Text and Images
- Stable Diffusion Image Variations