EEP 596: LLMs: From Transformers to GPT || Lecture 14

Dr. Karthik Mohan

Univ. of Washington, Seattle

February ~~23~~ 22, 2024

Deep Learning and Transformers References

Deep Learning

Great reference for the theory and fundamentals of deep learning: Book by Goodfellow and Bengio et al Bengio et al

Deep Learning History

Embeddings

SBERT and its usefulness

SBert Details

Instacart Search Relevance

Instacart Auto-Complete

Attention

Illustration of attention mechanism

Generative AI References

Prompt Engineering

Prompt Design and Engineering: Introduction and Advanced Methods

Retrieval Augmented Generation (RAG)

Toolformer

RAG Toolformer explained

Misc GenAI references

Time-Aware Language Models as Temporal Knowledge Bases

Generative AI references

Stable Diffusion

The Original Stable Diffusion Paper ✓ → Latent Diffusion Model

Reference: CLIP → Image + Text Joint Embedding (OpenAI)

Diffusion Explainer: Visual Explanation for Text-to-image Stable Diffusion

The Illustrated Stable Diffusion

Web Demo of Stable Diffusion

Pretty good visual of SD

Previous Lecture

- Auto Encoder and their use-cases
- De-noising Auto-Encoder

This Lecture

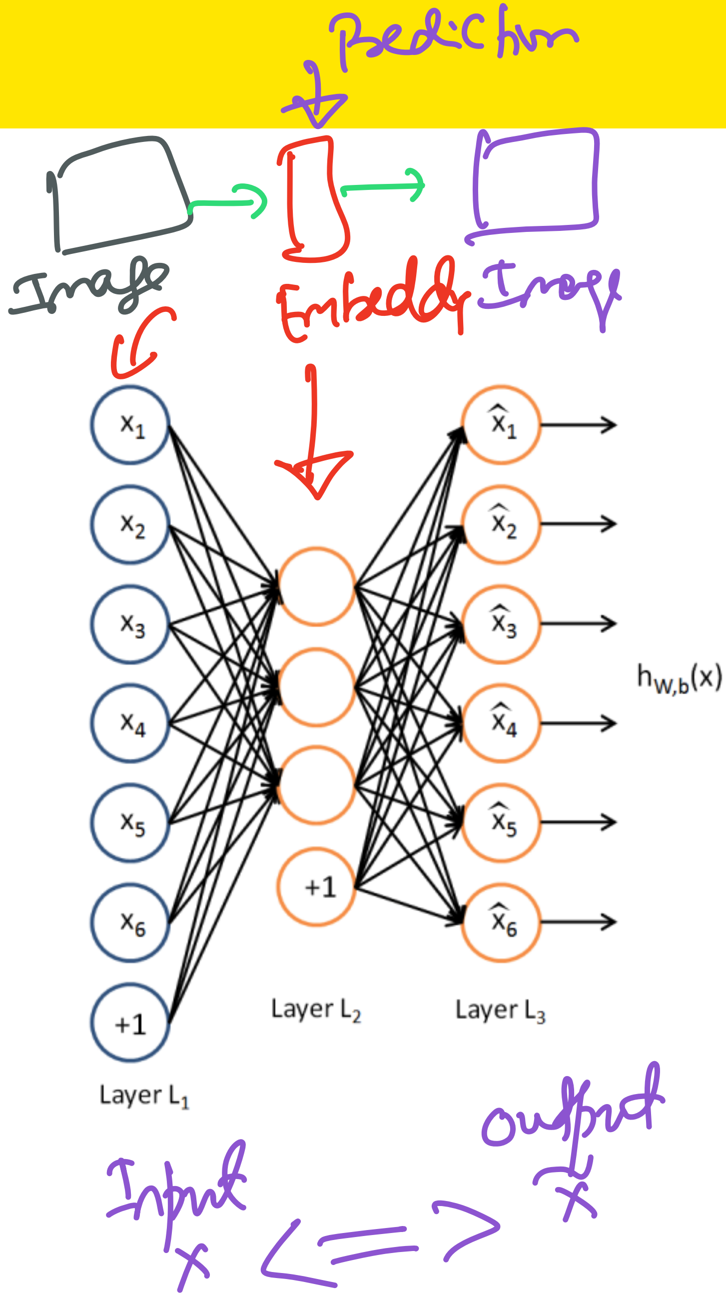
- Stable Diffusion Architecture
- De-noising AutoEncoders in Stable Diffusion

Stable Diffusion Explained

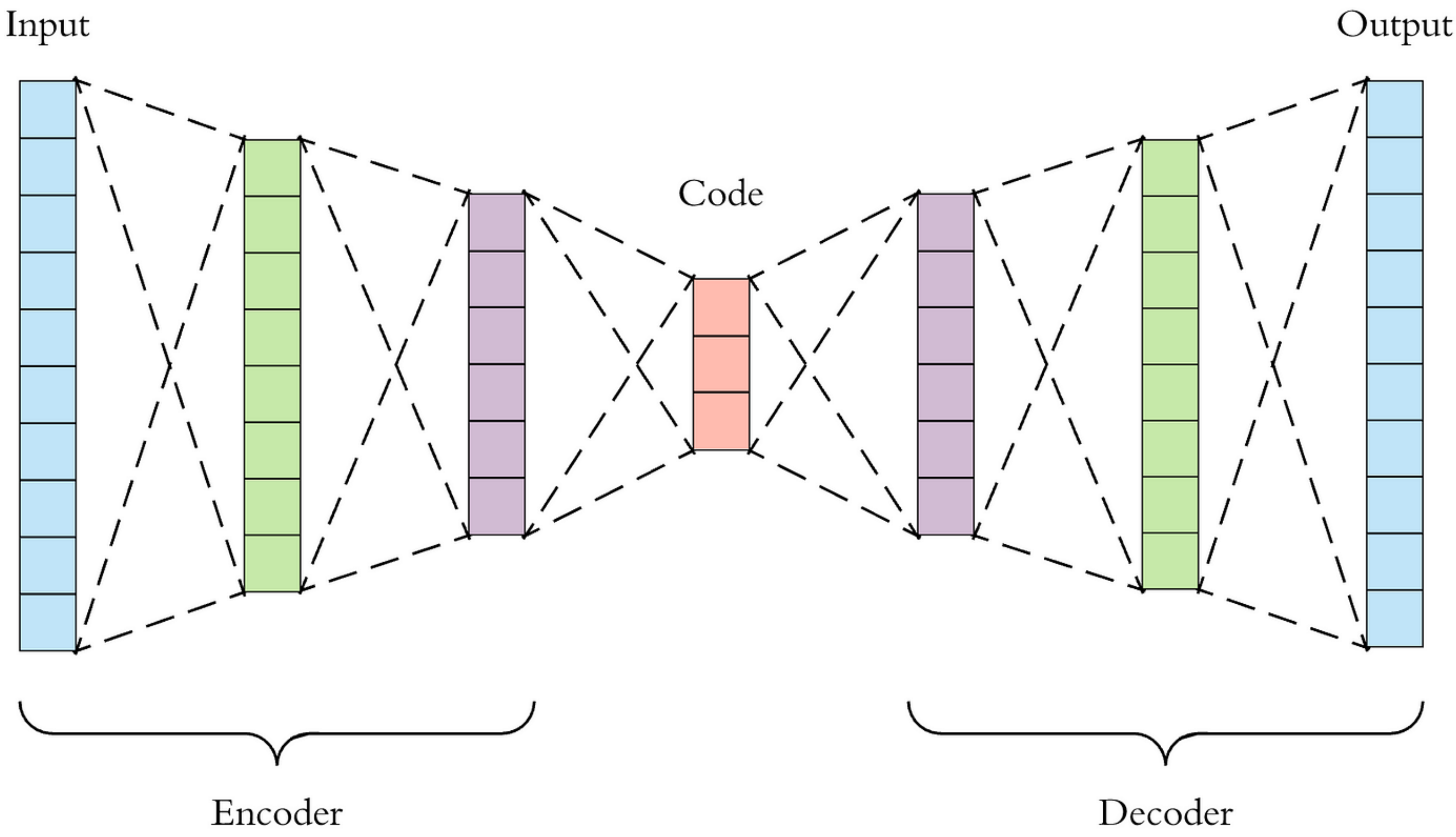
→ successive de-noising!

- Based on the concept of "de-noising auto encoders" and the use of text prompt to *guide the de-noising*
- Stable diffusion is also trained to successfully de-noise and increase the resolution of the image using text guidance

Auto Encoders



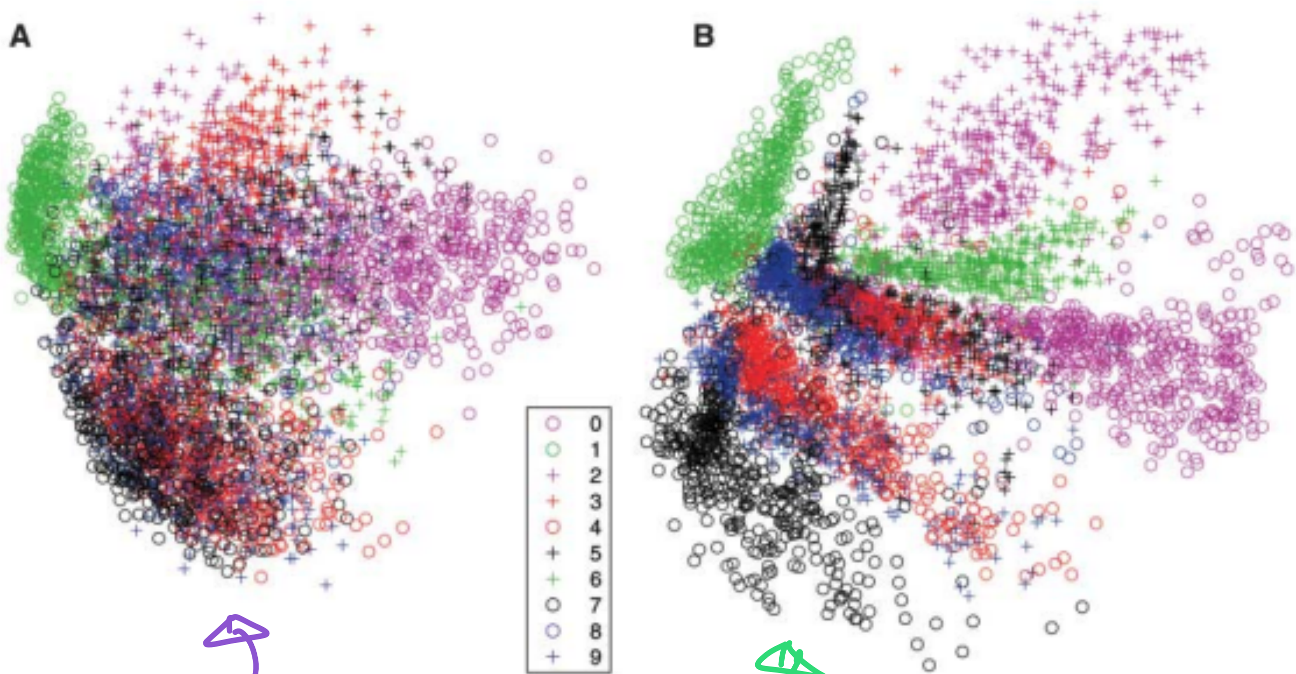
Deep Auto Encoders



AutoEncoders and Dimensionality Reduction

Reading Reference for AE Dimensionality Reduction

Fig. 3. (**A**) The two-dimensional codes for 500 digits of each class produced by taking the first two principal components of all 60,000 training images. (**B**) The two-dimensional codes found by a 784-1000-500-250-2 autoencoder. For an alternative visualization, see (*8*).



PCA

AE

AutoEncders Summary

1. Auto-Encoders are a method for dimensionality reduction and can do better than PCA for visualization

AutoEncders Summary

1. Auto-Encoders are a method for dimensionality reduction and can do better than PCA for visualization
2. Use Neural Networks architecture and hence can encode non-linearity in the embeddings

Depth + Non-Linearity

AutoEncders Summary

1. Auto-Encoders are a method for dimensionality reduction and can do better than PCA for visualization
2. Use Neural Networks architecture and hence can encode non-linearity in the embeddings
3. Anything else?

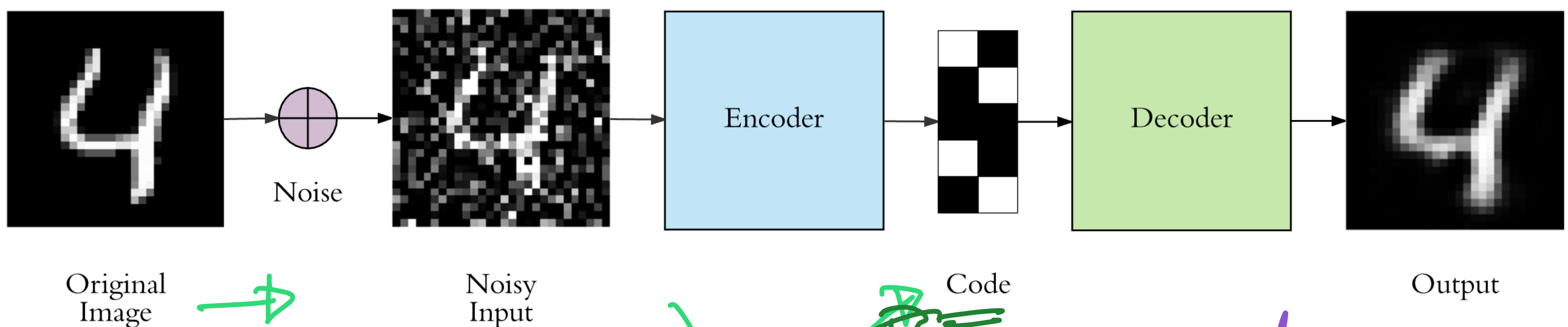
AutoEncders Summary

1. Auto-Encoders are a method for dimensionality reduction and can do better than PCA for visualization
2. Use Neural Networks architecture and hence can encode non-linearity in the embeddings
3. Anything else?
4. Auto Encoders can learn convolutional layers instead of dense layers - Better for images! More flexibility!!

De-noising Auto Encoders

"Handling Images in Latent Space"

Stable Diffusion ⟺ Diffusion in Latent Space!

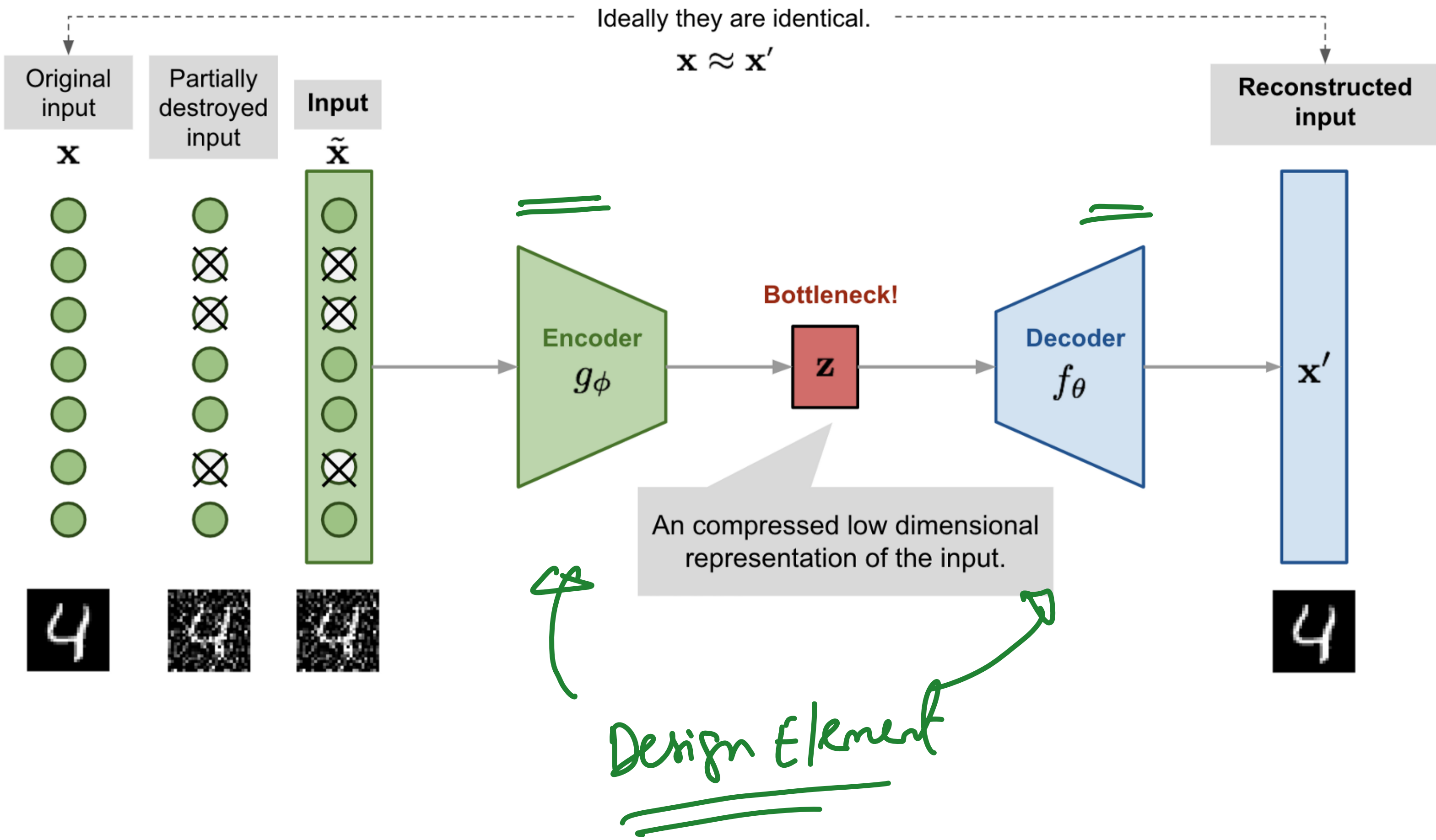


Embeddings
→128 dim
embeddings
→128 dim.

vector space
→Latent space

Latent Representation
or
Latent Space → space of Embeddings
=
Embedding
=
Encoding = Code

De-noising Auto Encoders



De-noising Auto Encoders

Business Context

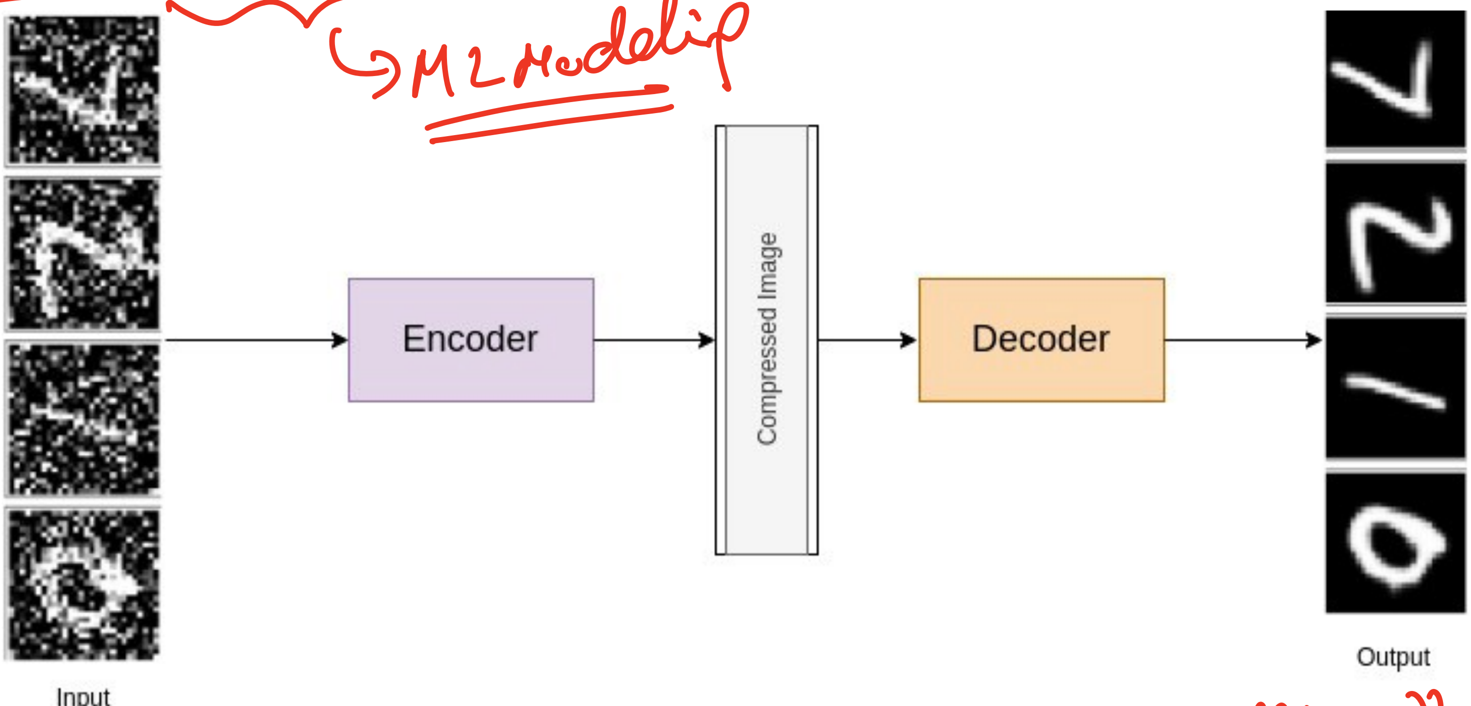
Data

1. Business Problem → 2. ML/AI Task → 3. Loss function + 4. Architecture

↳ 5. Train model

↳ 6. Prediction

↳ ML modeling



Loss functions depend on the "task"

Define Task ⇒ Design Loss function

De-noising Auto Encoders

Details

- Just like an Auto Encoder

De-noising Auto Encoders

Details

- Just like an Auto Encoder
- Difference: Noise is injected in the inputs on purpose but output is a clean data point.

1. Robustness } In a future lecture!
2. mimicking real data/applications

Adversarial testing of models

ML models

LLMs

De-noising Auto Encoders

Details

- Just like an Auto Encoder
- Difference: Noise is injected in the inputs on purpose but output is a clean data point.
- This forces the Auto Encoder to “de-noise” data, esp. useful for images!

De-noising Auto Encoders

Details

- Just like an Auto Encoder
- Difference: Noise is injected in the inputs on purpose but output is a clean data point.
- This forces the Auto Encoder to “de-noise” data, esp. useful for images!
- Esp. useful for a category of objects or images (e.g. digit recognition or face recognition, etc)

Stable Diffusion High-Level

1. only text

Black Box

"Text → Image"

Reference: The Illustrated Stable Diffusion

LOGISTICS

1. Mini-Project 3

Component 1

Improve GPT 3.5 using "Toolformer" style approach

(APE)

Component 2

use stability AI / Stable Diffusion APIs to solve simple tasks of "image editing"

2. Final Presentation

Finals week
T & Th 4PM – 5:45PM

7 minutes

5 + 1 + 1 → Q&A

RAG

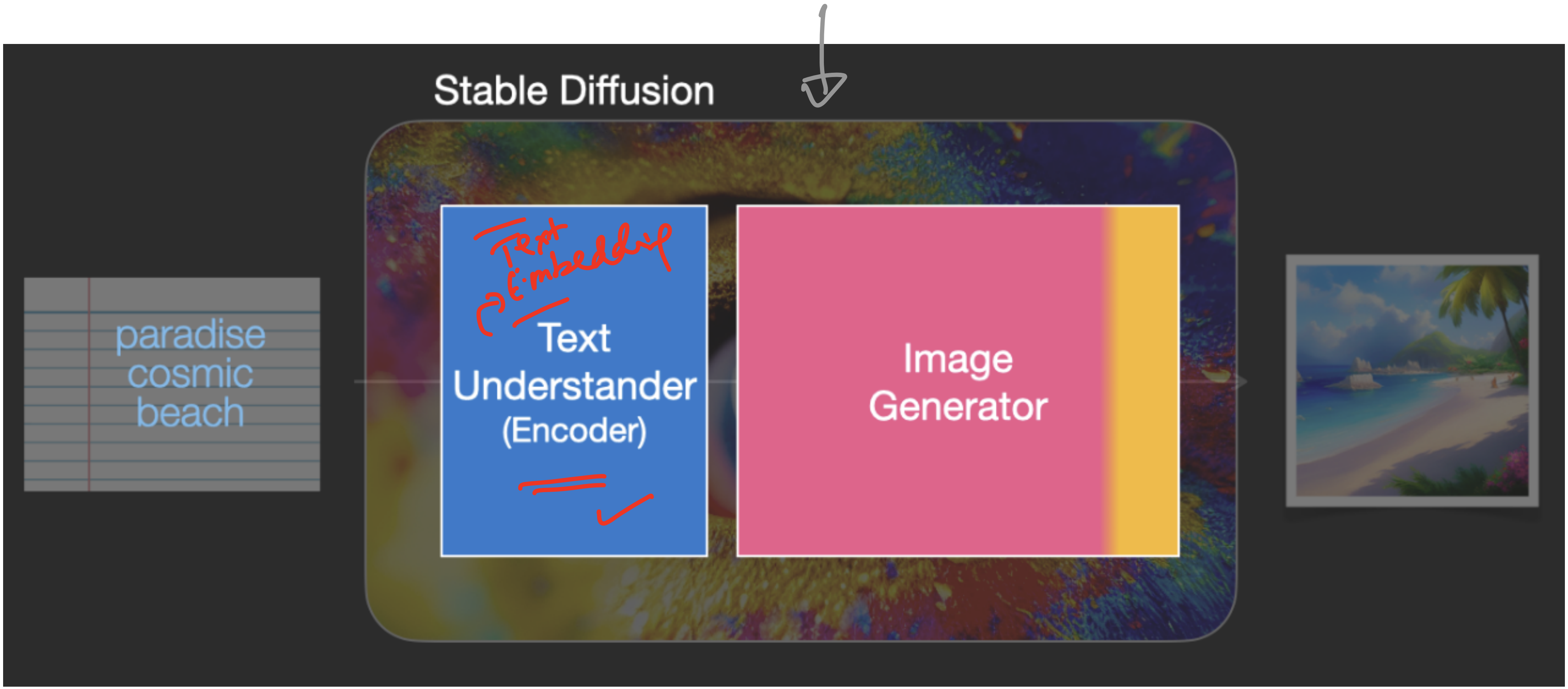
Calculator GPT 3.5 / Stable Diffusion

Stable Diffusion High-Level

2) Text + Image → SD → Enhanced Image

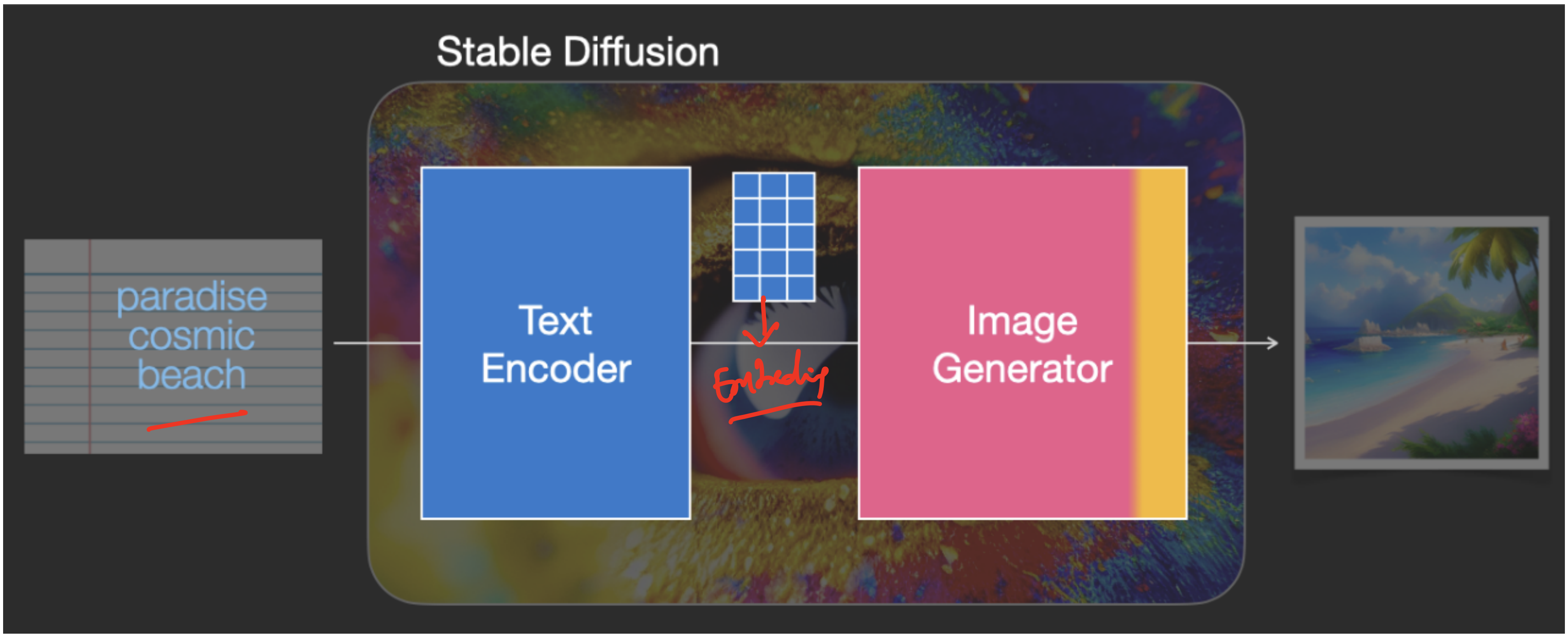
Reference: The Illustrated Stable Diffusion

Stable Diffusion Components



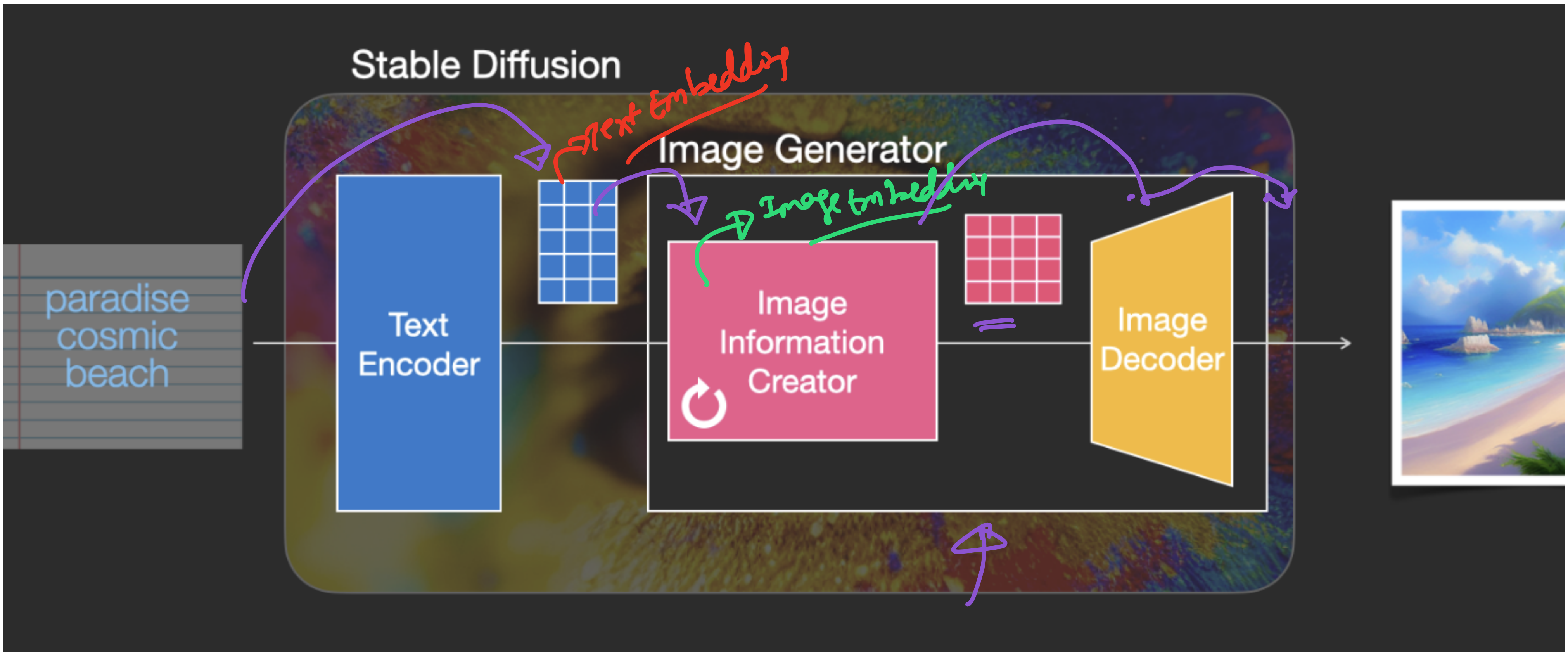
Reference: The Illustrated Stable Diffusion

Stable Diffusion Components



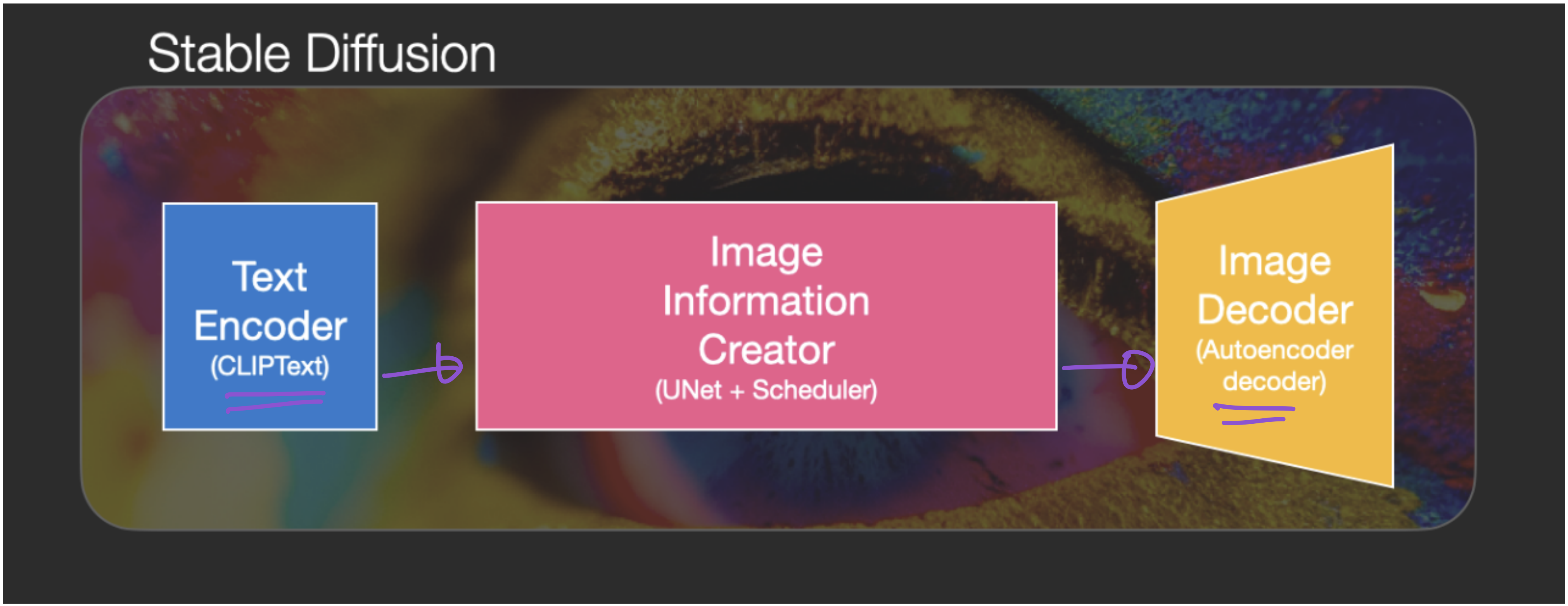
Reference: The Illustrated Stable Diffusion

Image Information Creator



Reference: The Illustrated Stable Diffusion

Image Decoder

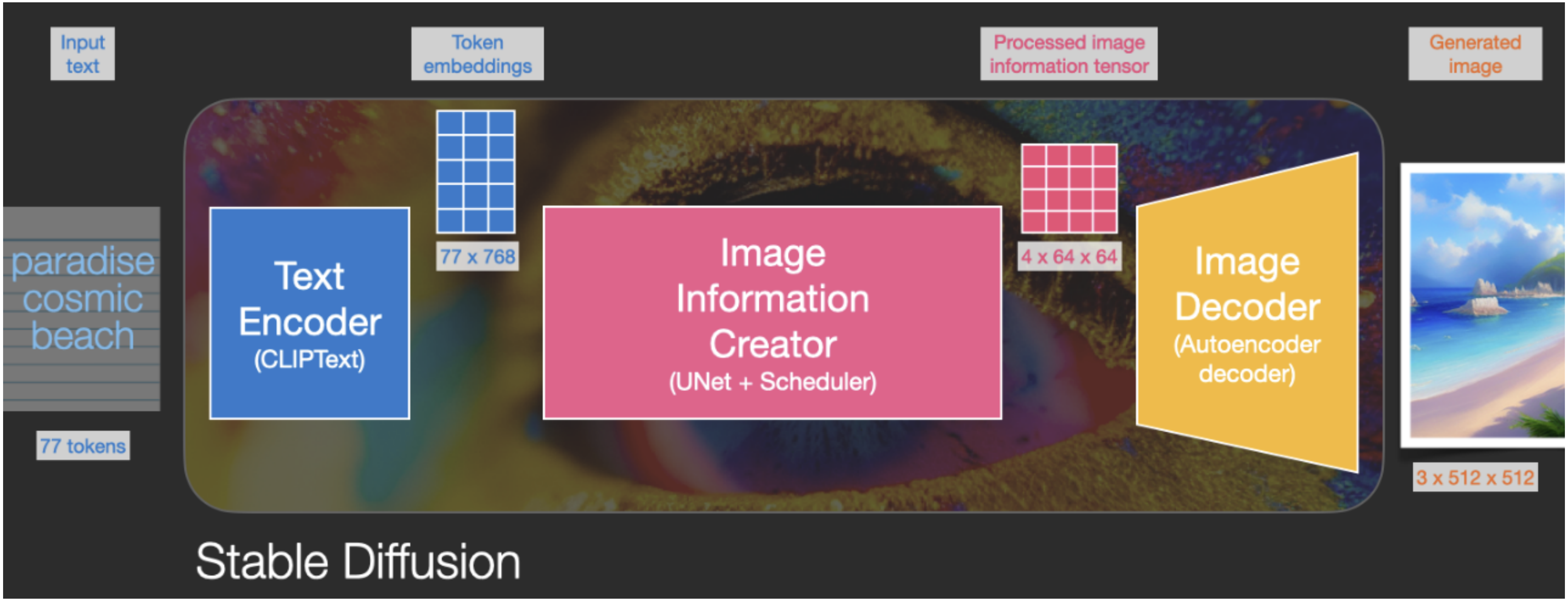


Reference: The Illustrated Stable Diffusion

Stable Diffusion - Break down of components

1. **Text Encoding:** Uses ClipText
2. **Image diffusion process:** Take in a text embedding and generate an image embedding that can then be converted to an image.
3. **Image Decoder:** This is an AutoEncoder-Decoder that takes in an Image embedding and returns an image in a pixel format (512x512x3)

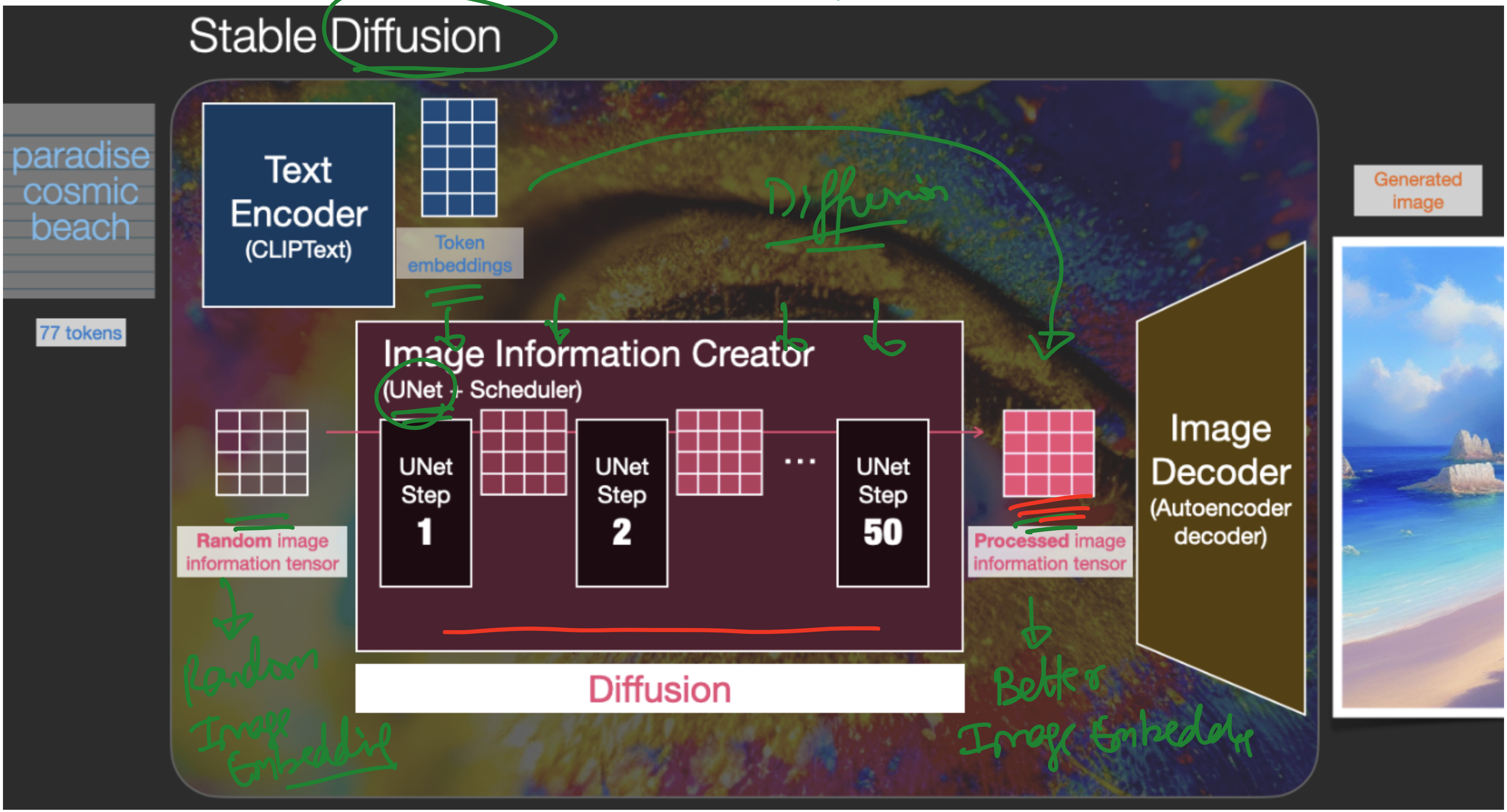
High-level



Reference: The Illustrated Stable Diffusion

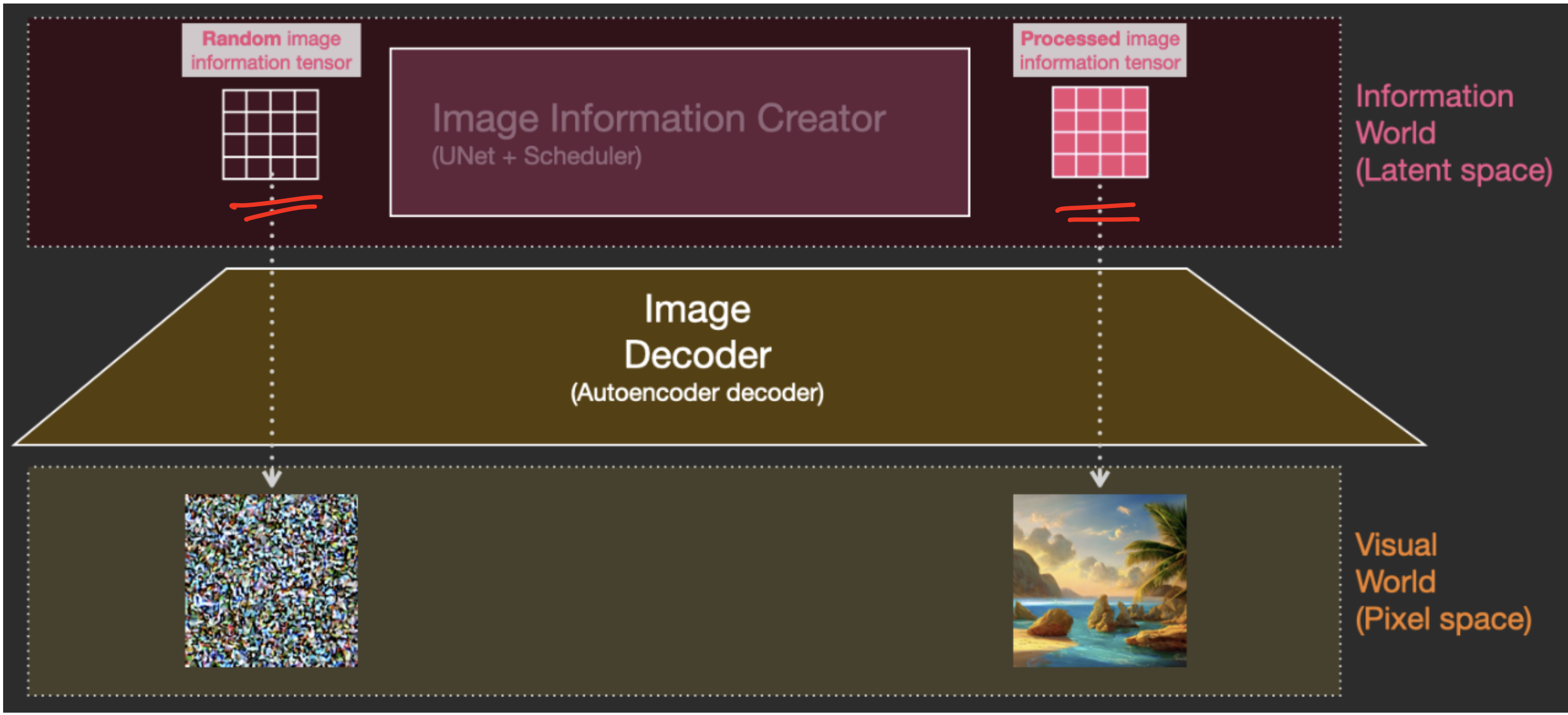
Understanding Diffusion

Text Embed → Image Embed (Diffusion) UNET – DE-NOISING AE for Images



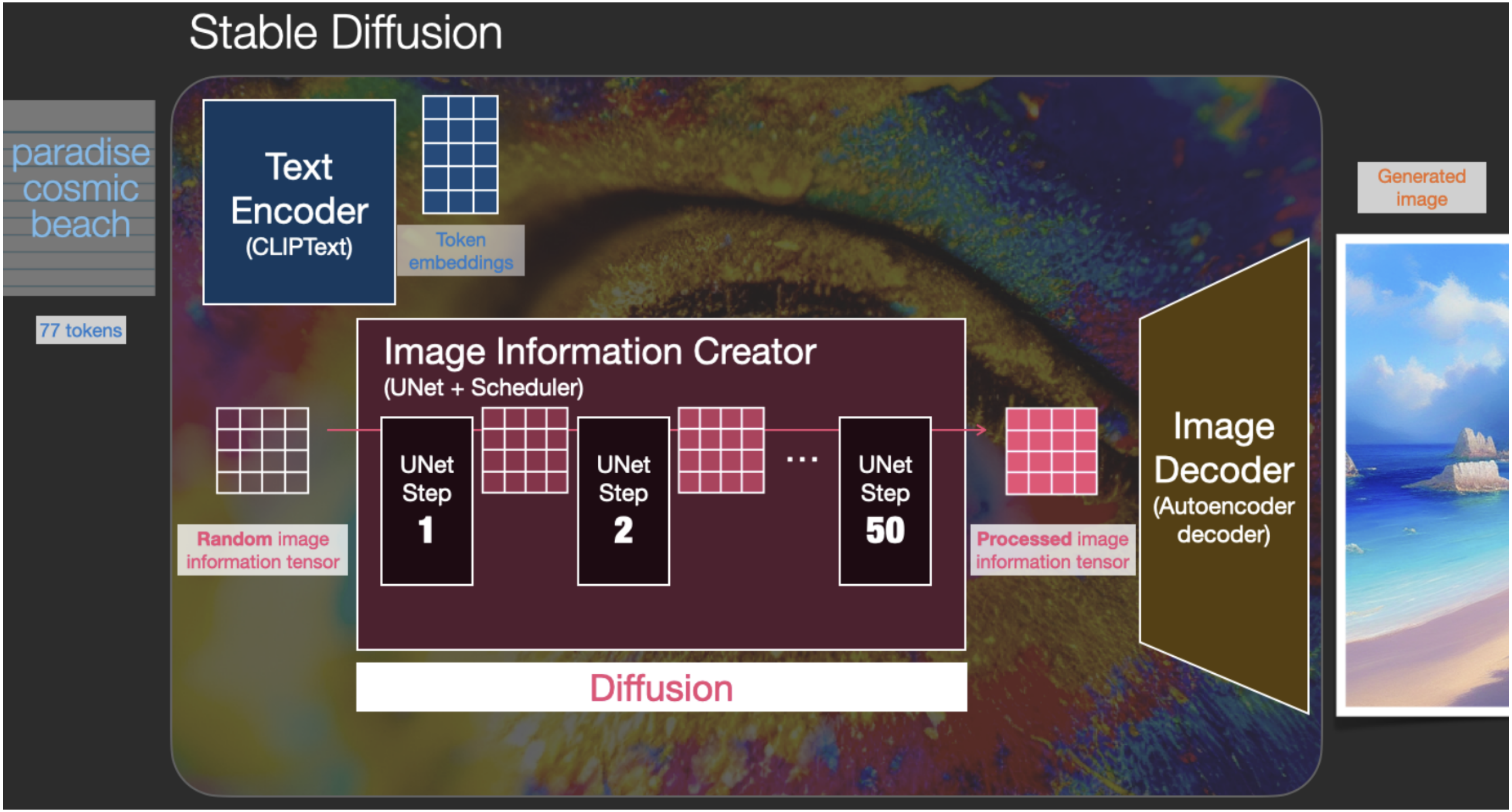
Reference: The Illustrated Stable Diffusion

Understanding Diffusion



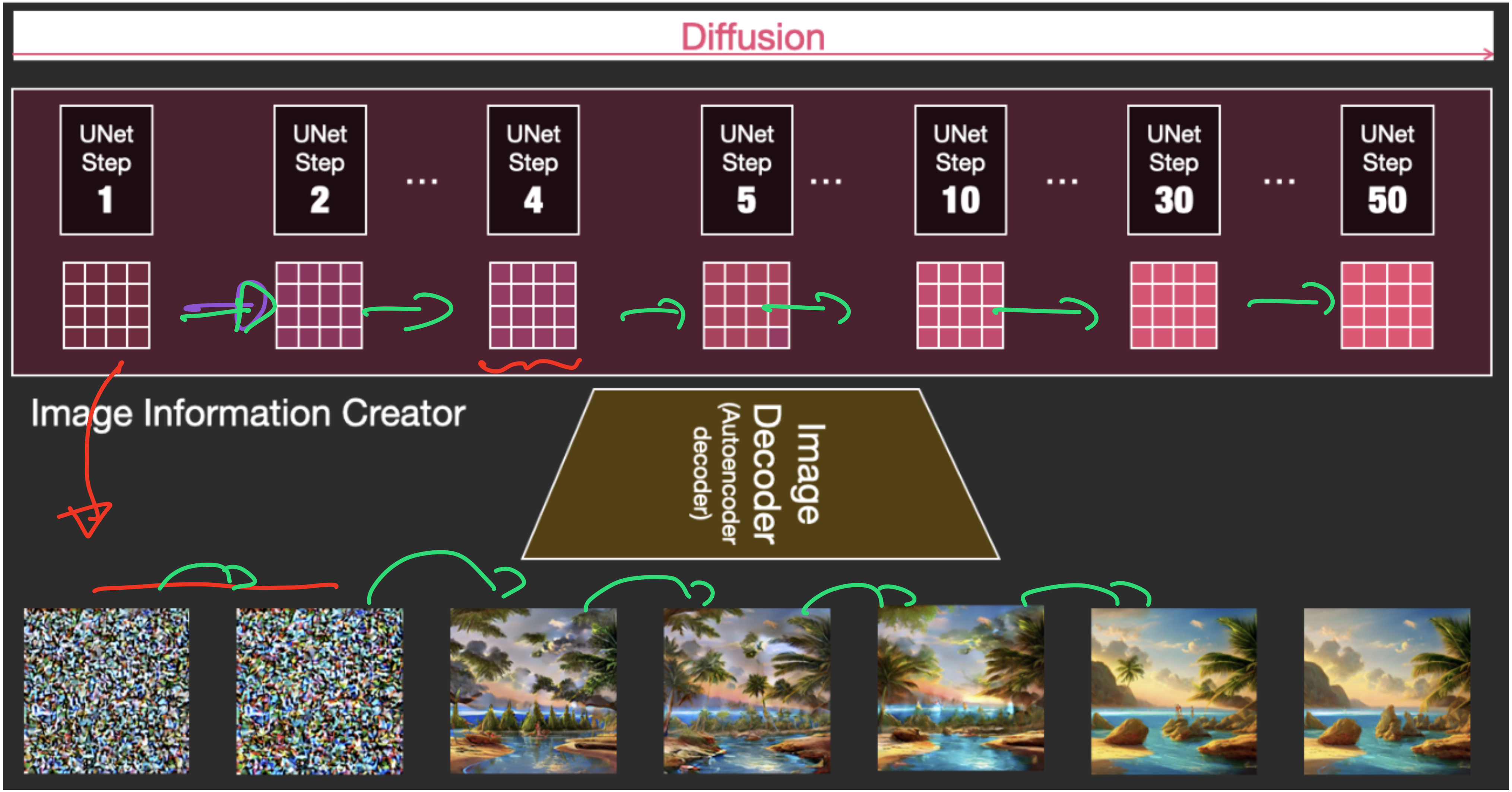
Reference: The Illustrated Stable Diffusion

Understanding Diffusion



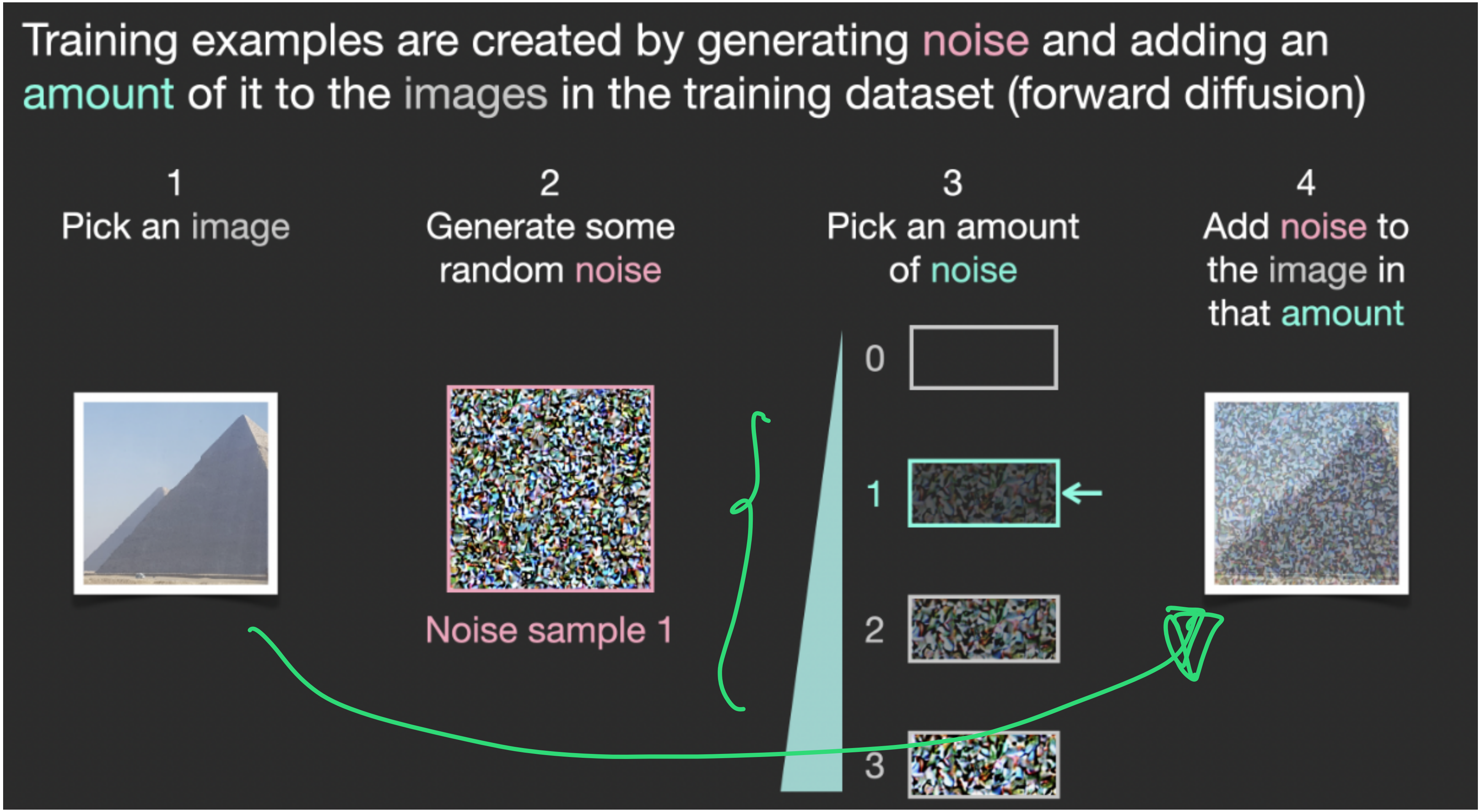
Reference: The Illustrated Stable Diffusion

Understanding Diffusion



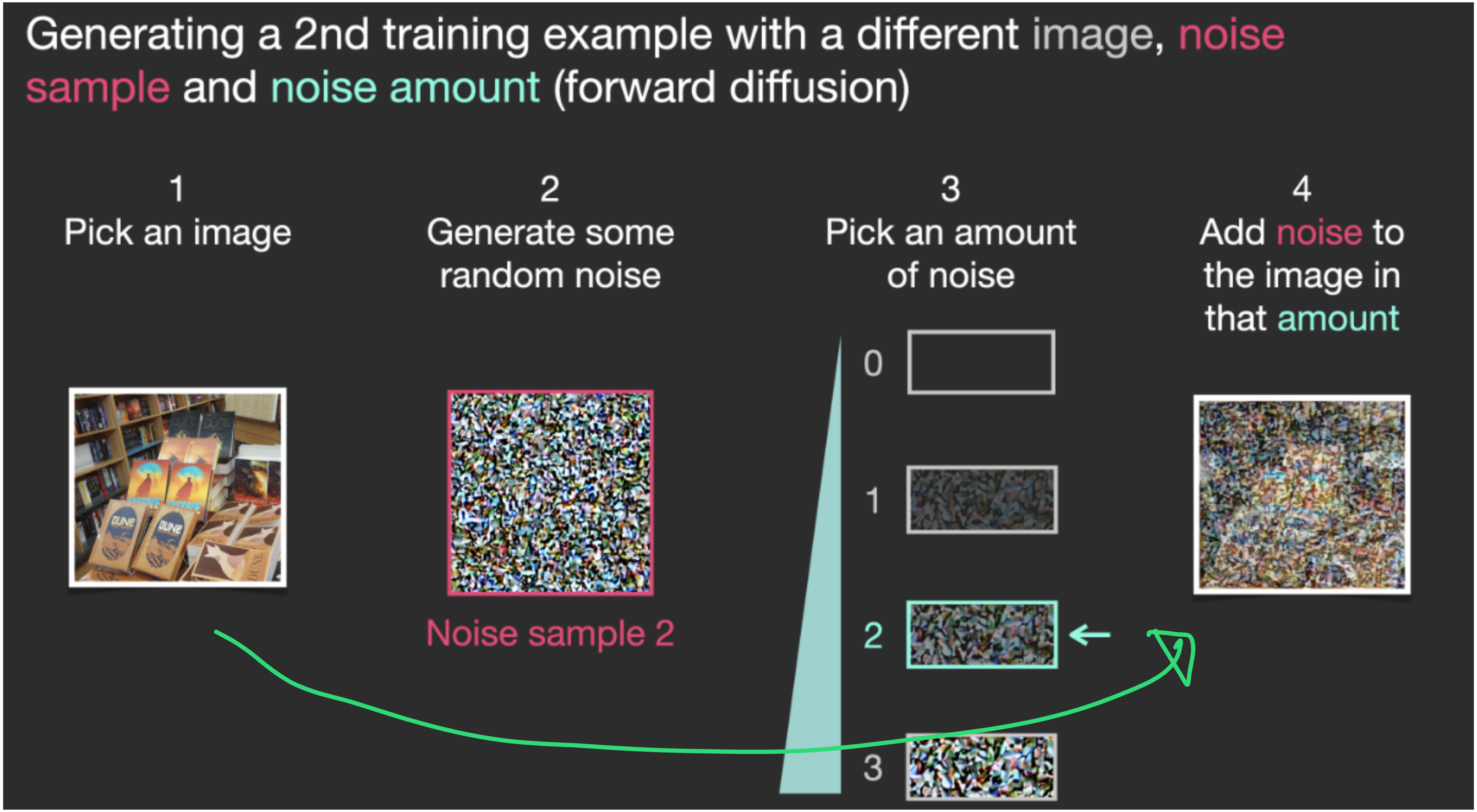
Reference: The Illustrated Stable Diffusion

Generating Training Examples with Noise Addition



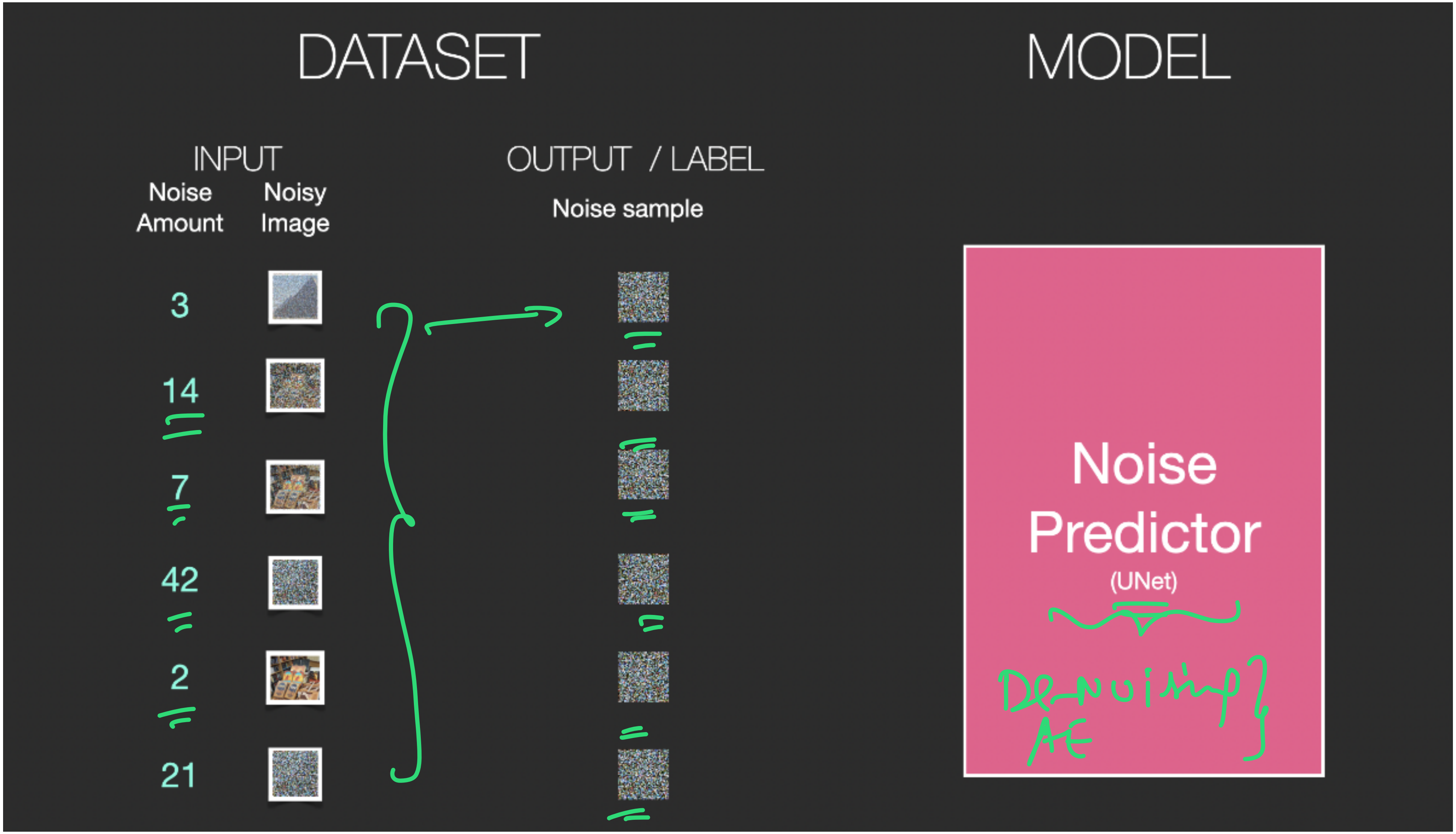
Reference: The Illustrated Stable Diffusion

Generating Training Examples with Noise Addition



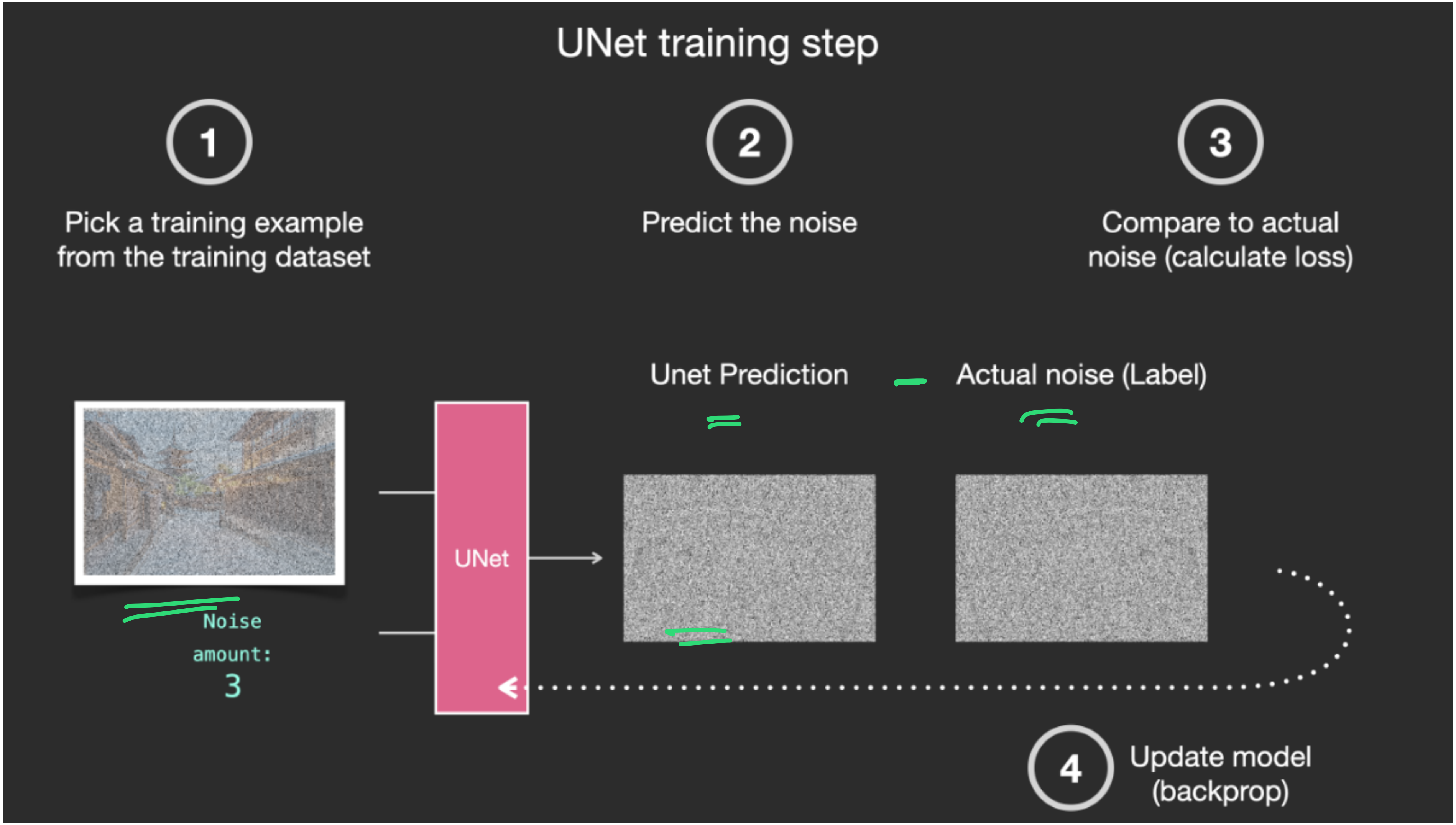
Reference: The Illustrated Stable Diffusion

Generating Training Examples with Noise Addition



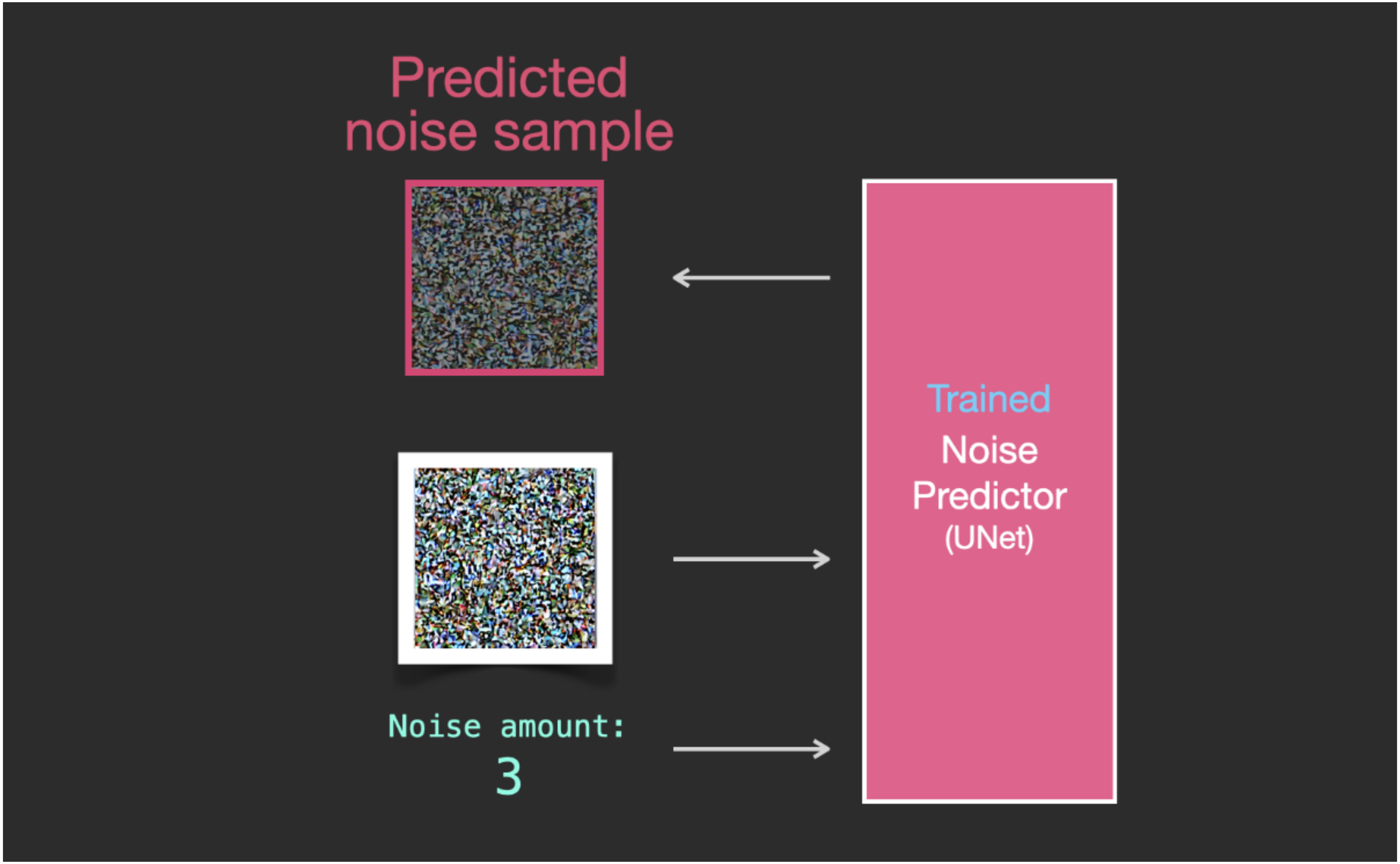
Reference: The Illustrated Stable Diffusion

Training Process



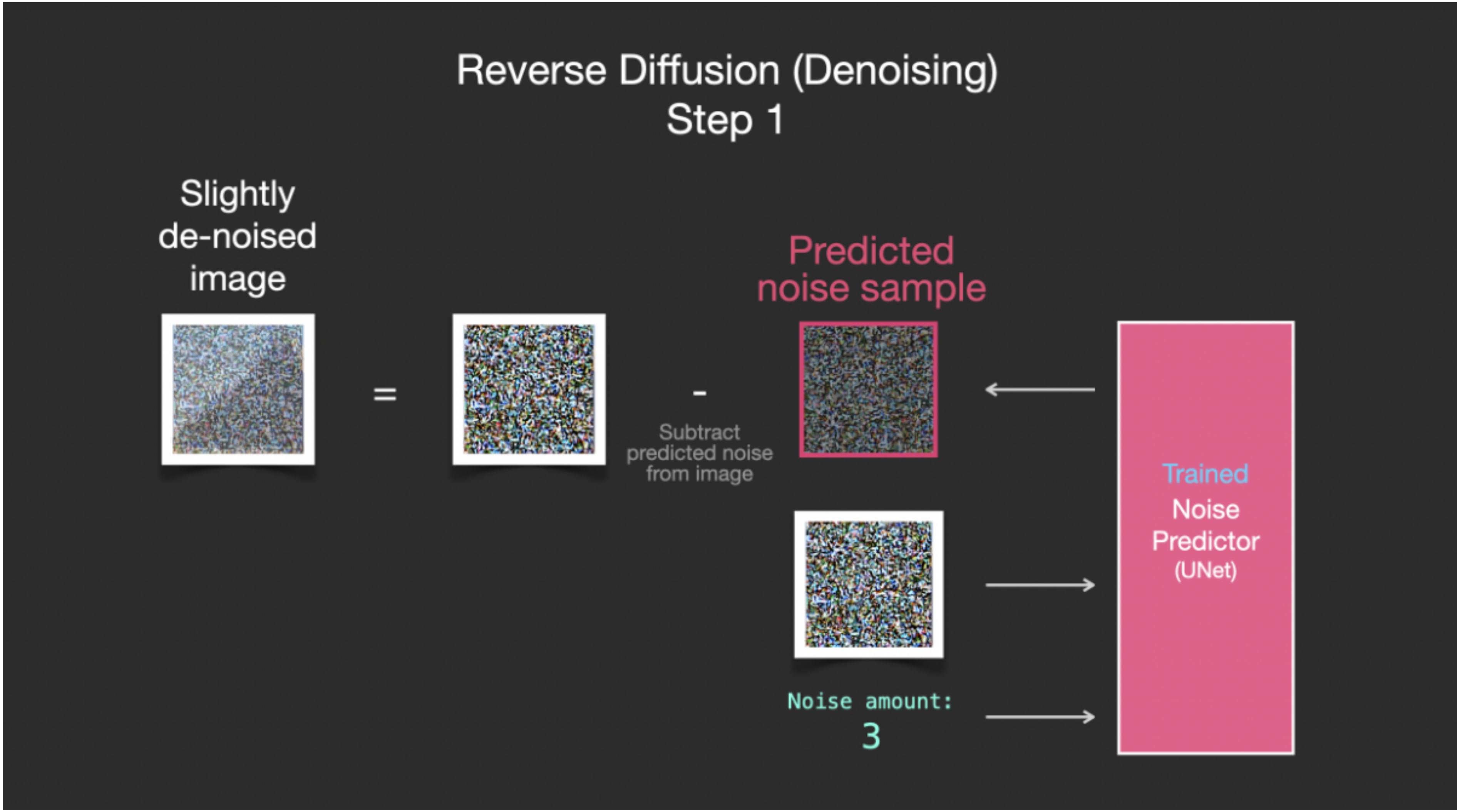
Reference: The Illustrated Stable Diffusion

Predicting noise and Noise Removal



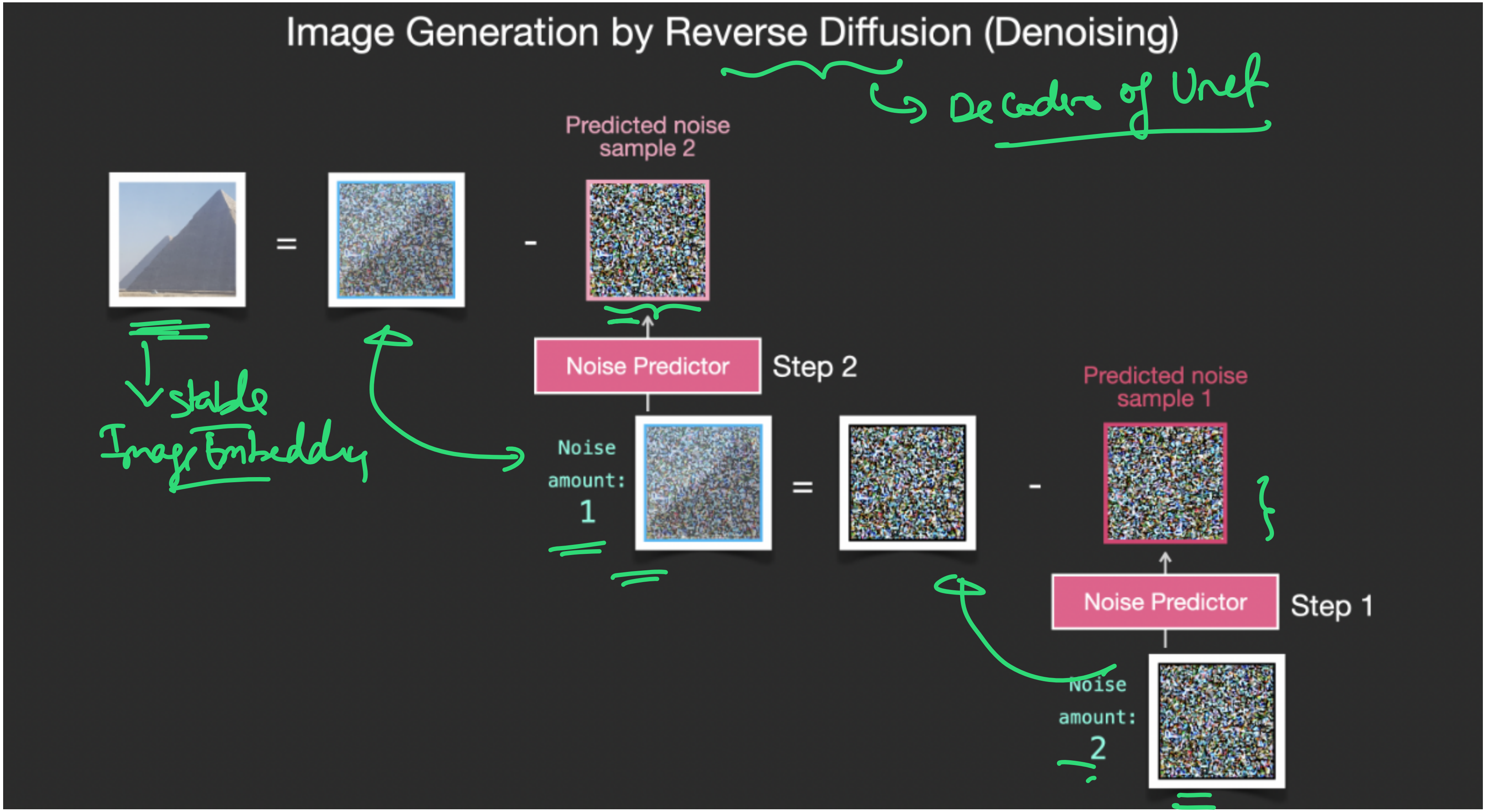
Reference: The Illustrated Stable Diffusion

Predicting noise and Noise Removal



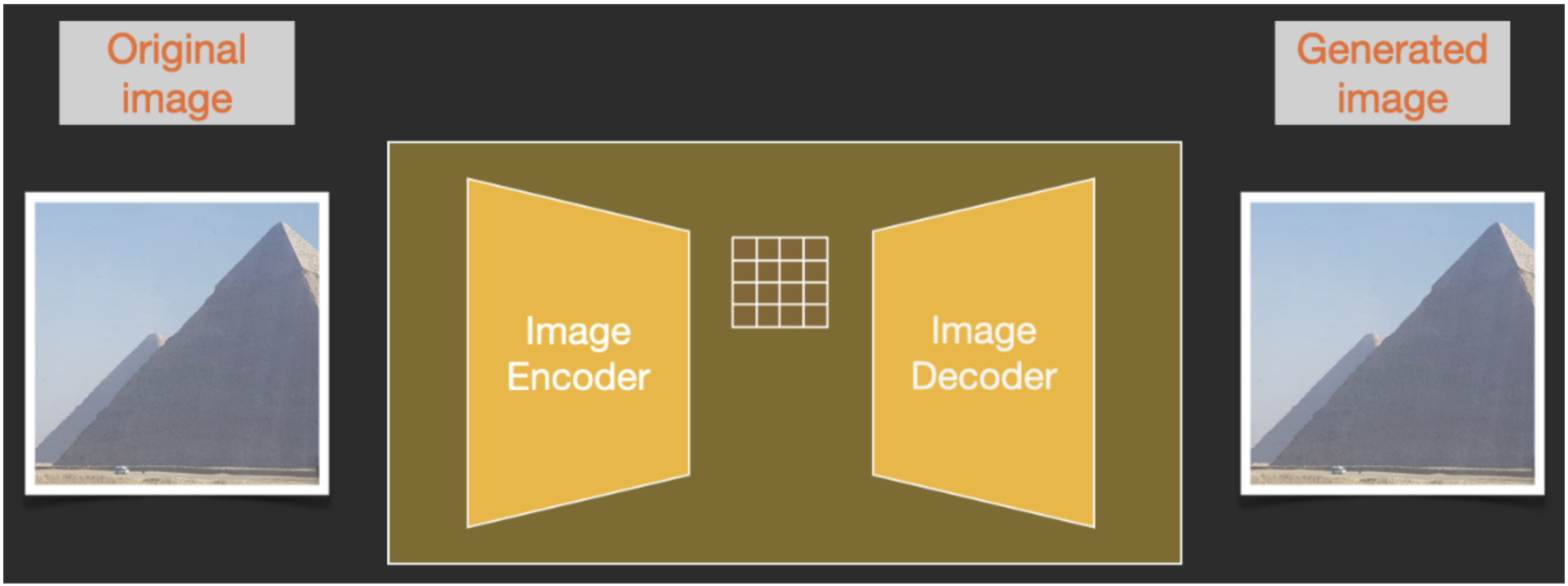
Reference: The Illustrated Stable Diffusion

Predicting noise and Noise Removal



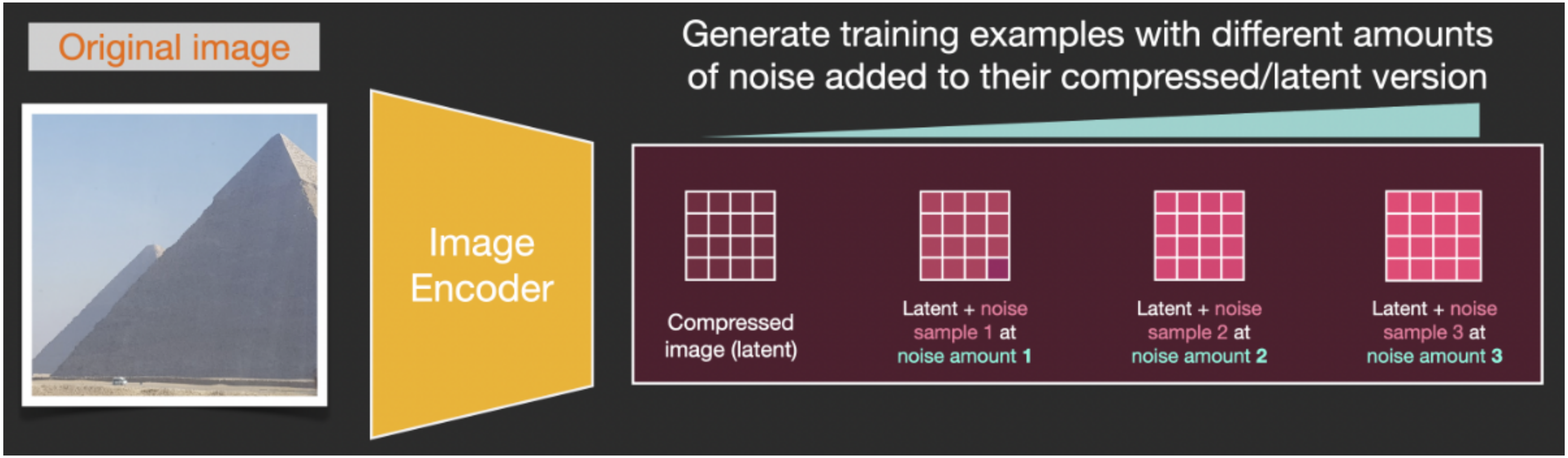
Reference: The Illustrated Stable Diffusion

Speeding up Stable Diffusion



Reference: The Illustrated Stable Diffusion

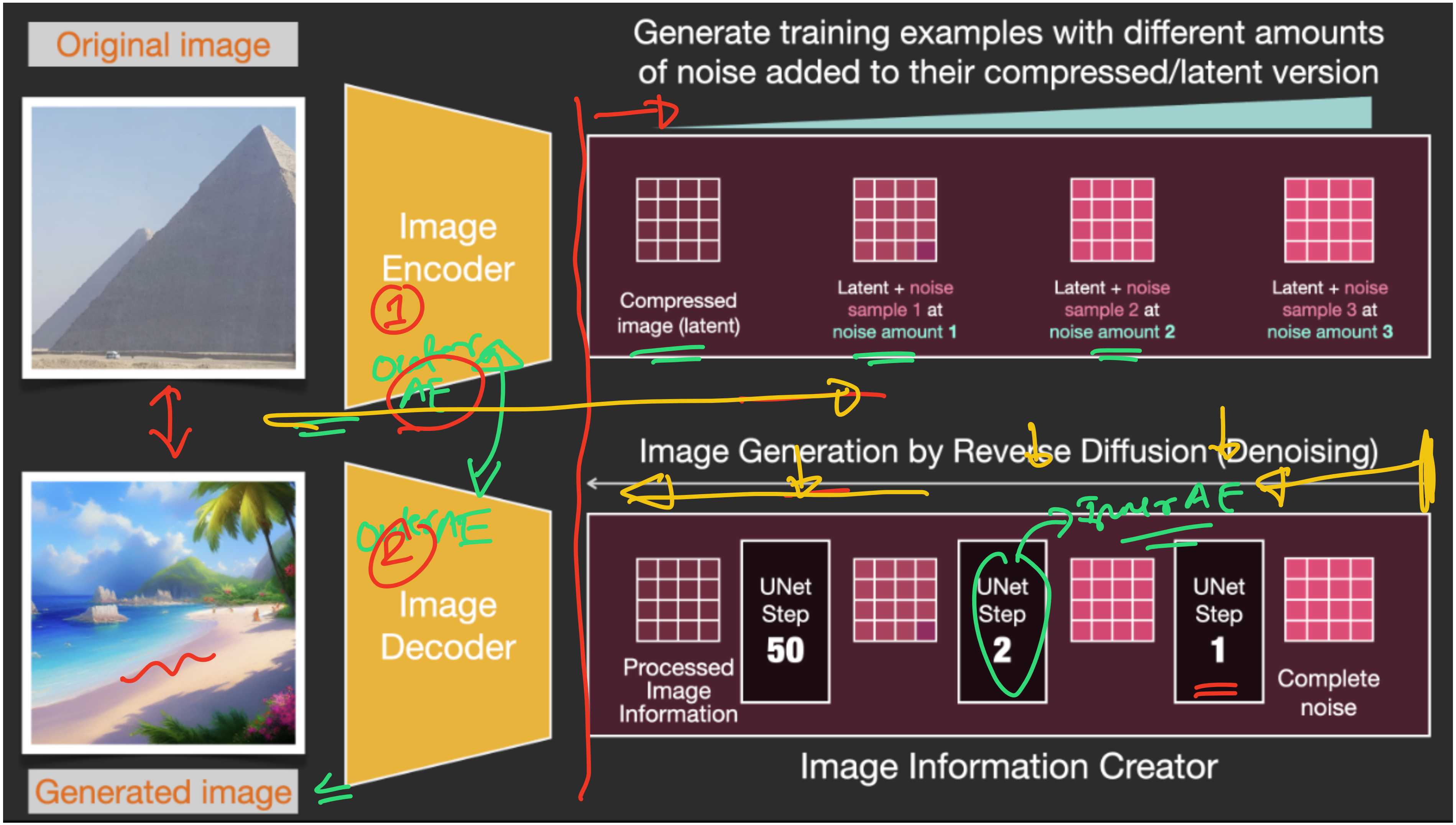
Speeding up Stable Diffusion



Reference: The Illustrated Stable Diffusion

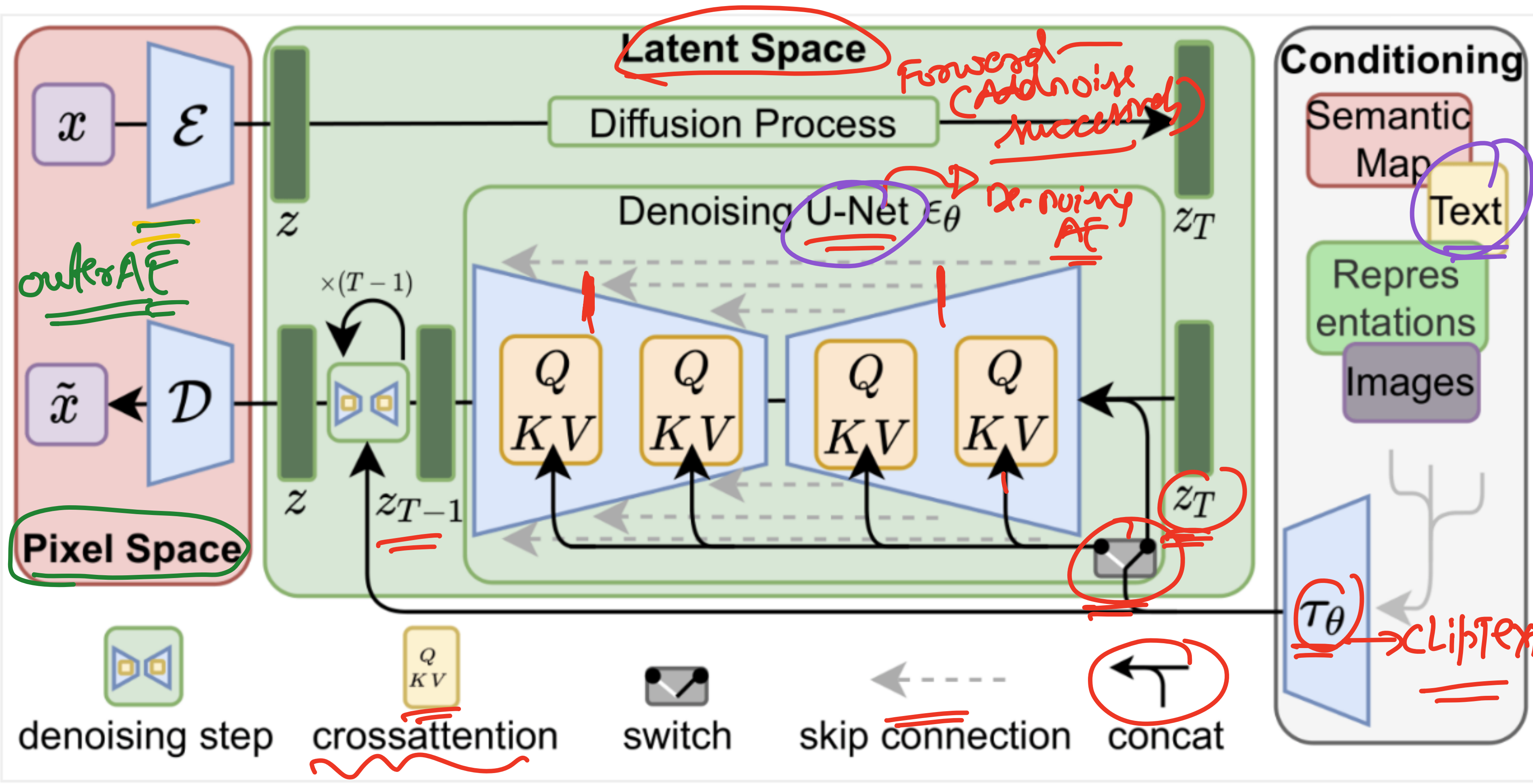
Speeding up Stable Diffusion

1 2 → outer AE



Reference: The Illustrated Stable Diffusion

Stable Diffusion Full Architecture



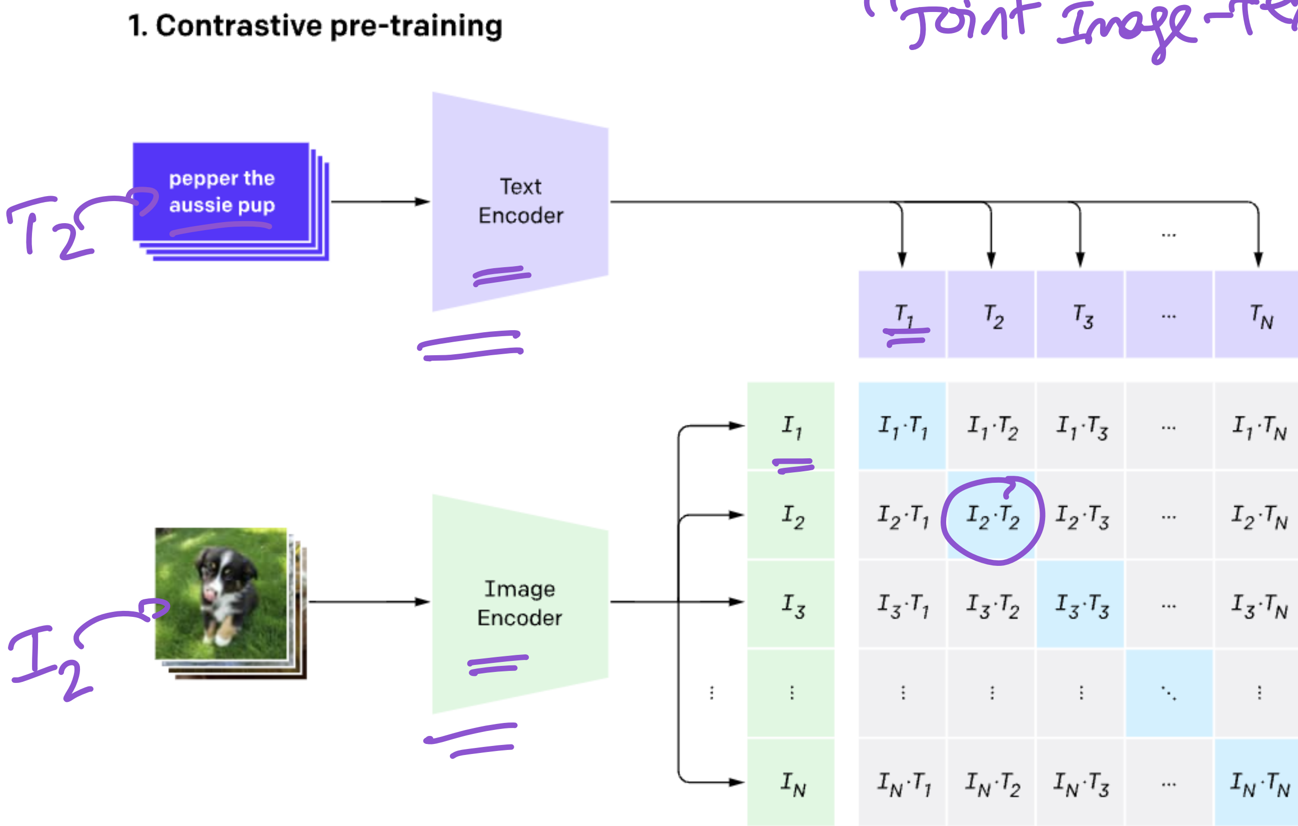
Reference: The Illustrated Stable Diffusion

Clip Pre-Training Architecture

OpenAI

("Joint Image-Text Latent Space")

Image Caption



$I_2 \leftrightarrow T_2$

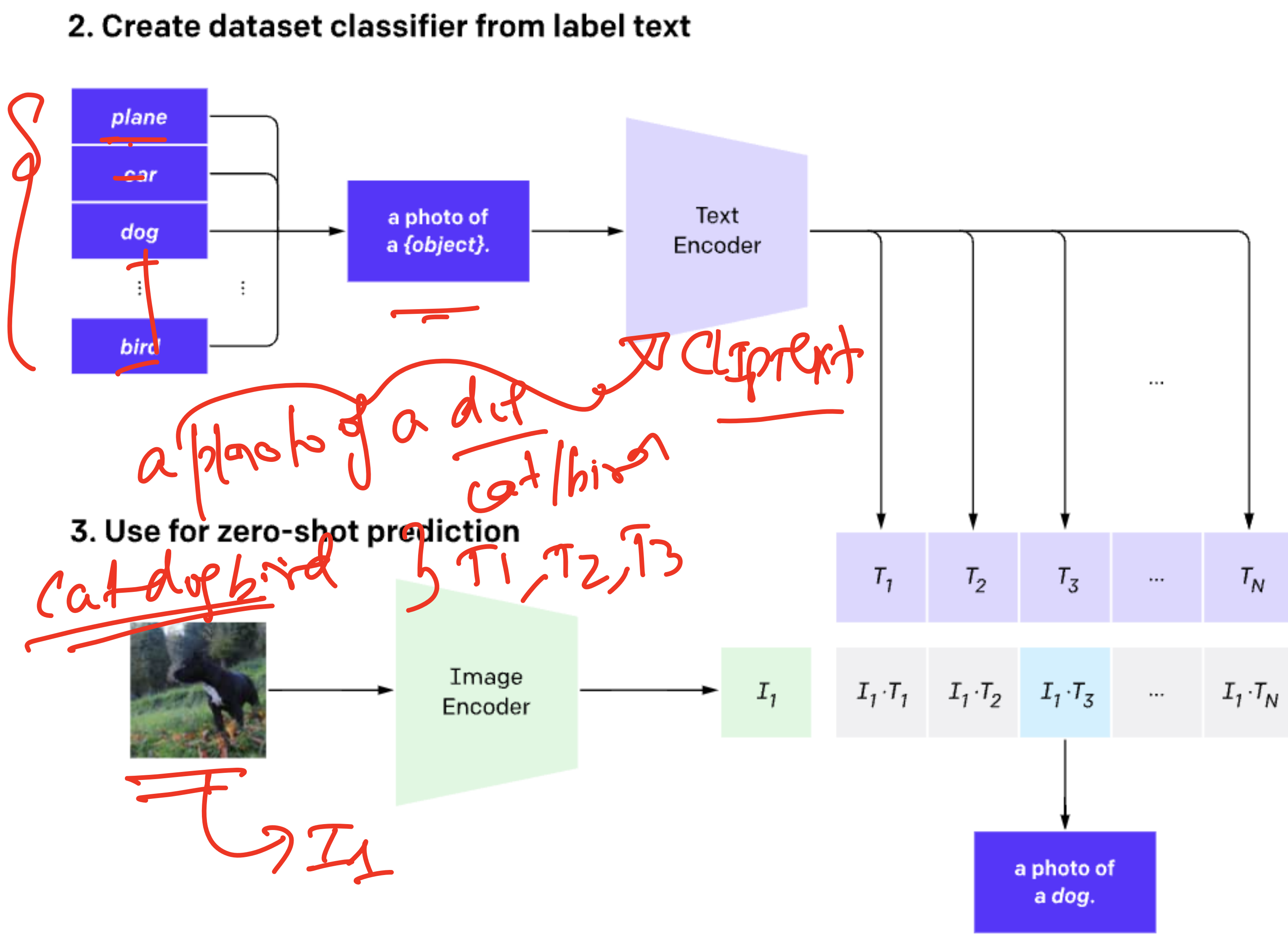
Predict 1 or -1 for a match

Cross-entropy for classification

CLIP pre-trains an image encoder and a text encoder to predict which images were paired with which texts in our dataset. We then use this behavior to turn CLIP into a zero-shot classifier. We convert all of a dataset's classes into captions such as "a photo of a dog" and predict the class of the caption CLIP estimates best pairs with a given image.

Reference: CLIP

Clip Zero-Shot Prediction Process



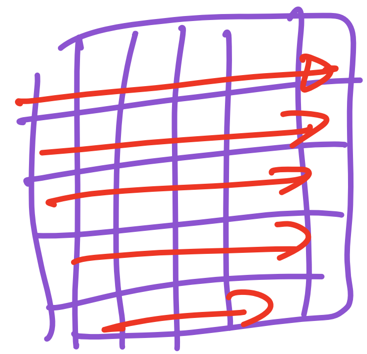
Reference: CLIP

ICE #1

What pre-trained encoders would CLIP probably have used for text and image encodings?

1. CNN for both
2. Word2Vec and CNN
3. Transformer and Vi Transformer
4. Glove and CNN

Vision Transformer



Clip Implementation Pseudo-code

vi transformer

```
# image_encoder - ResNet or Vision Transformer
# text_encoder  - CBOW or Text Transformer
# I[n, h, w, c] - minibatch of aligned images
# T[n, l]       - minibatch of aligned texts
# W_i[d_i, d_e] - learned proj of image to embed
# W_t[d_t, d_e] - learned proj of text to embed
# t             - learned temperature parameter

# extract feature representations of each modality
I_f = image_encoder(I) #[n, d_i]
T_f = text_encoder(T)  #[n, d_t]

# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)

# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)

# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axis=0)
loss_t = cross_entropy_loss(logits, labels, axis=1)
loss   = (loss_i + loss_t)/2
```

clip text

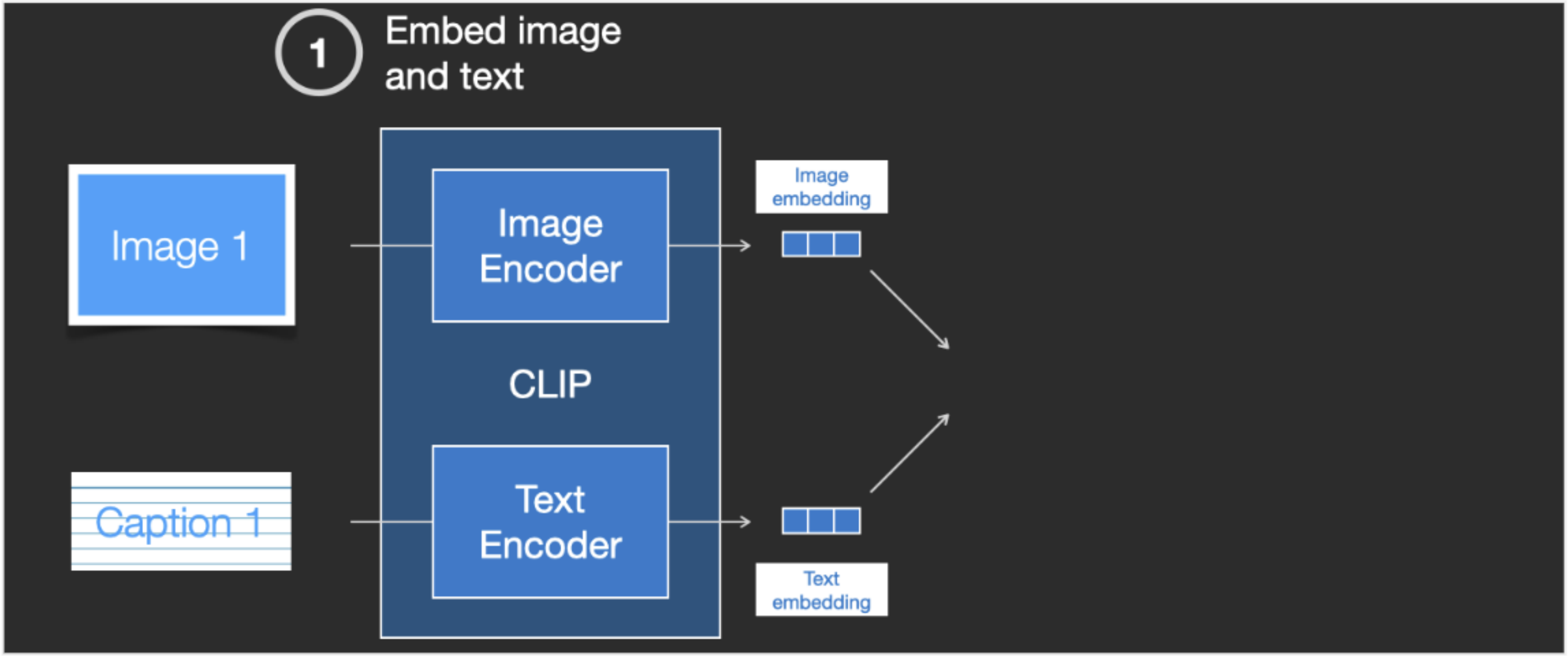
Figure 3. Numpy-like pseudocode for the core of an implementation of CLIP.

Reference: CLIP

Clip Training Examples

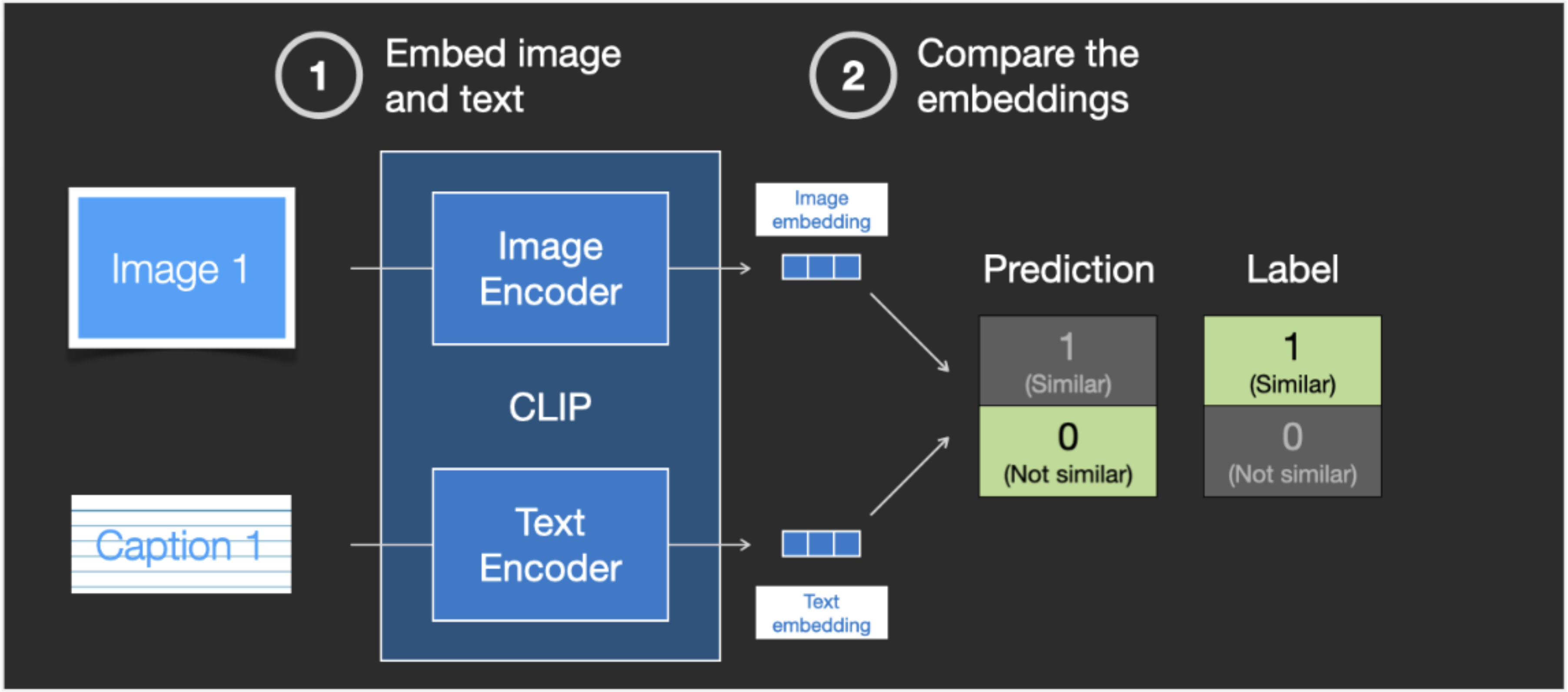
Question: How many examples used for Training? Reference: The Illustrated Stable Diffusion

Clip Training Process



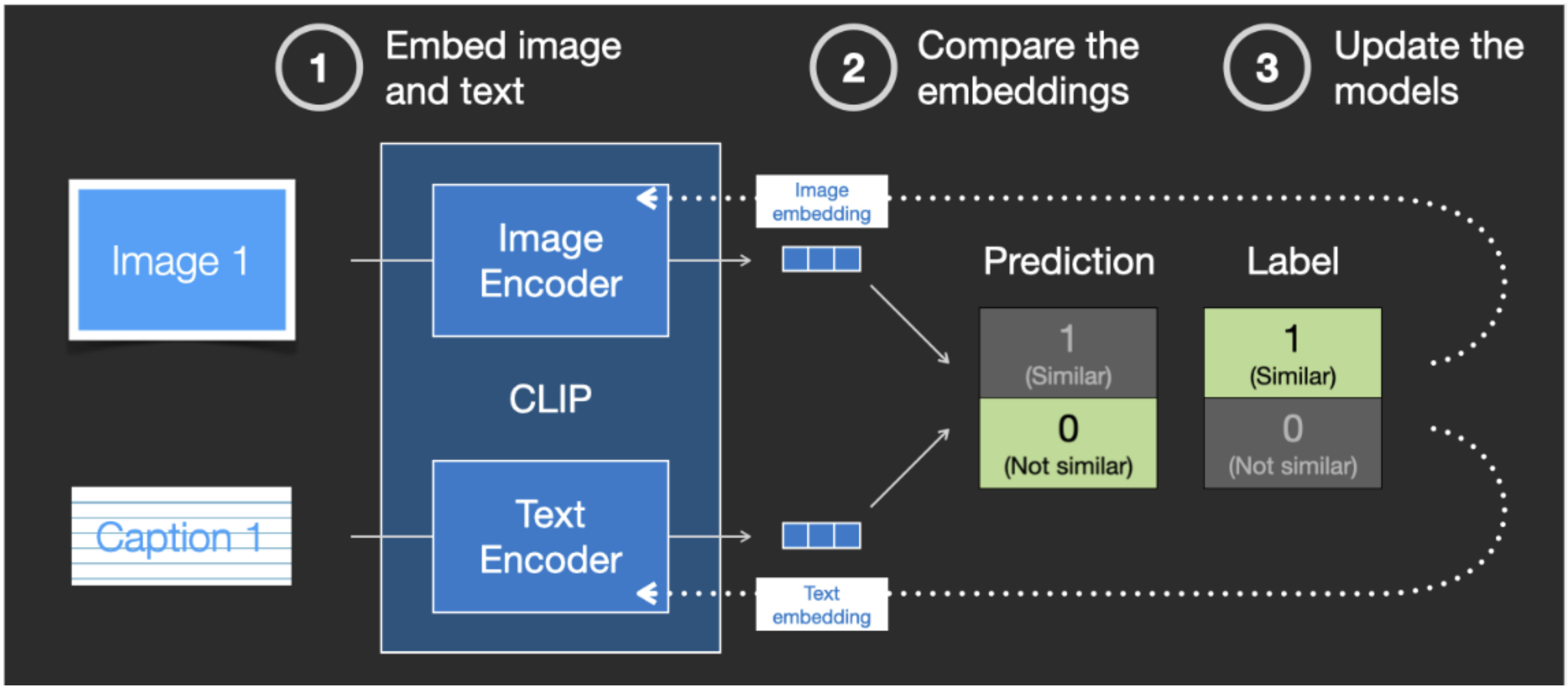
Reference: The Illustrated Stable Diffusion

Clip Training Process



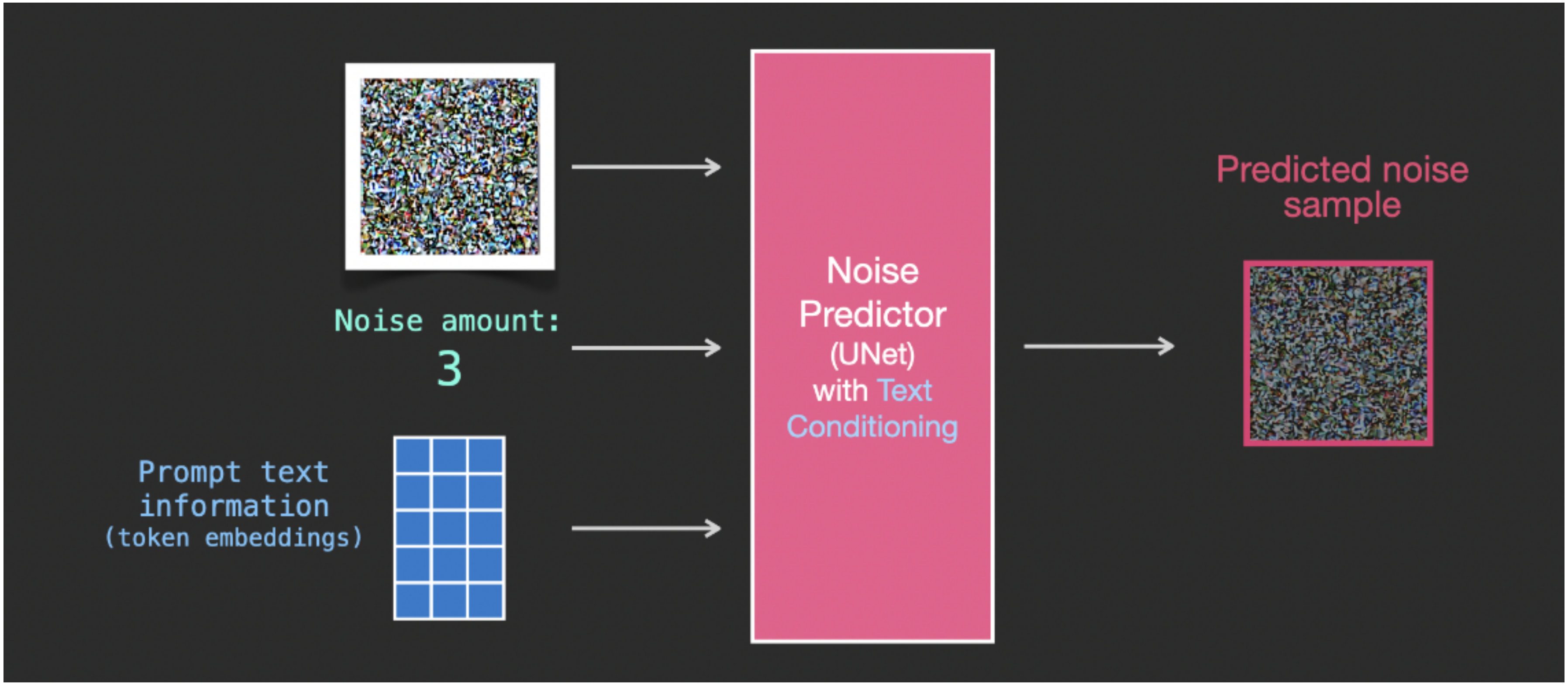
Reference: The Illustrated Stable Diffusion

Clip Training Process



Reference: The Illustrated Stable Diffusion

Image Generation Process

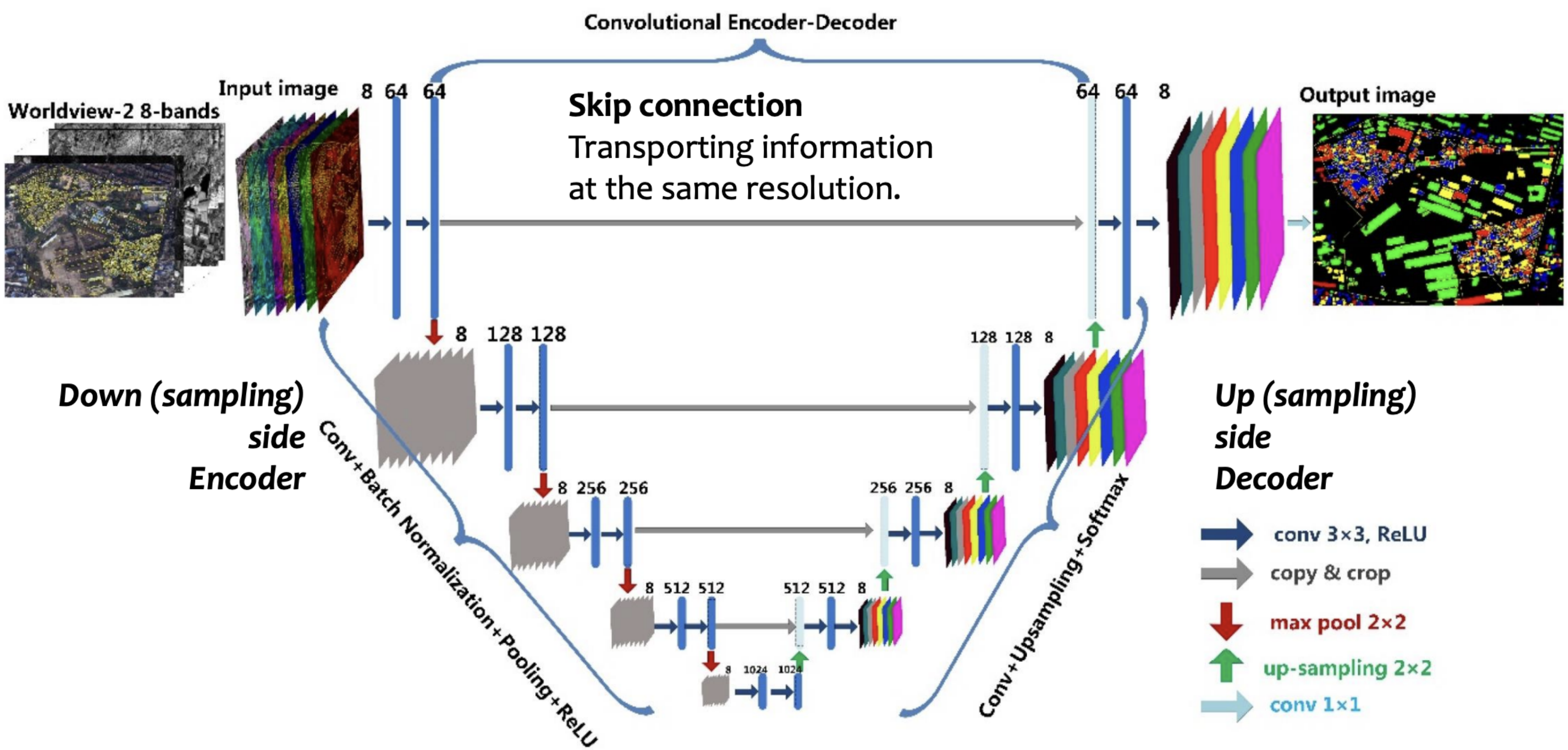


Reference: The Illustrated Stable Diffusion

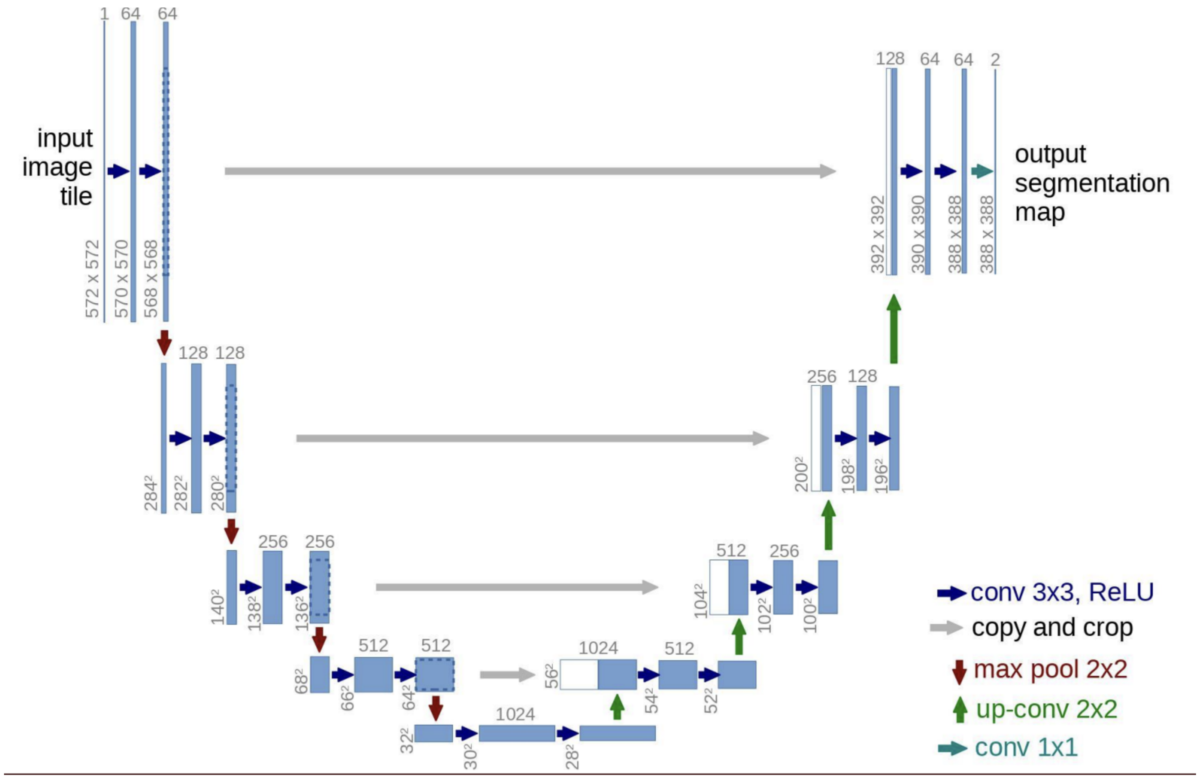
Image Generation: Training Data

Reference: The Illustrated Stable Diffusion

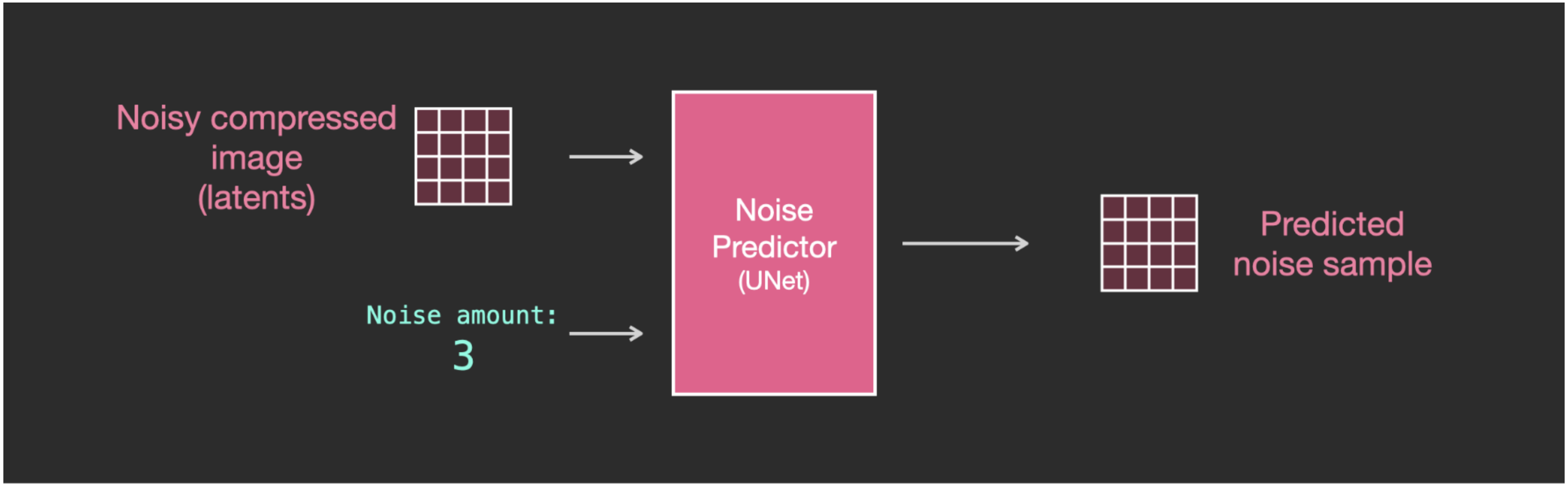
Unet Architecture



Unet Architecture

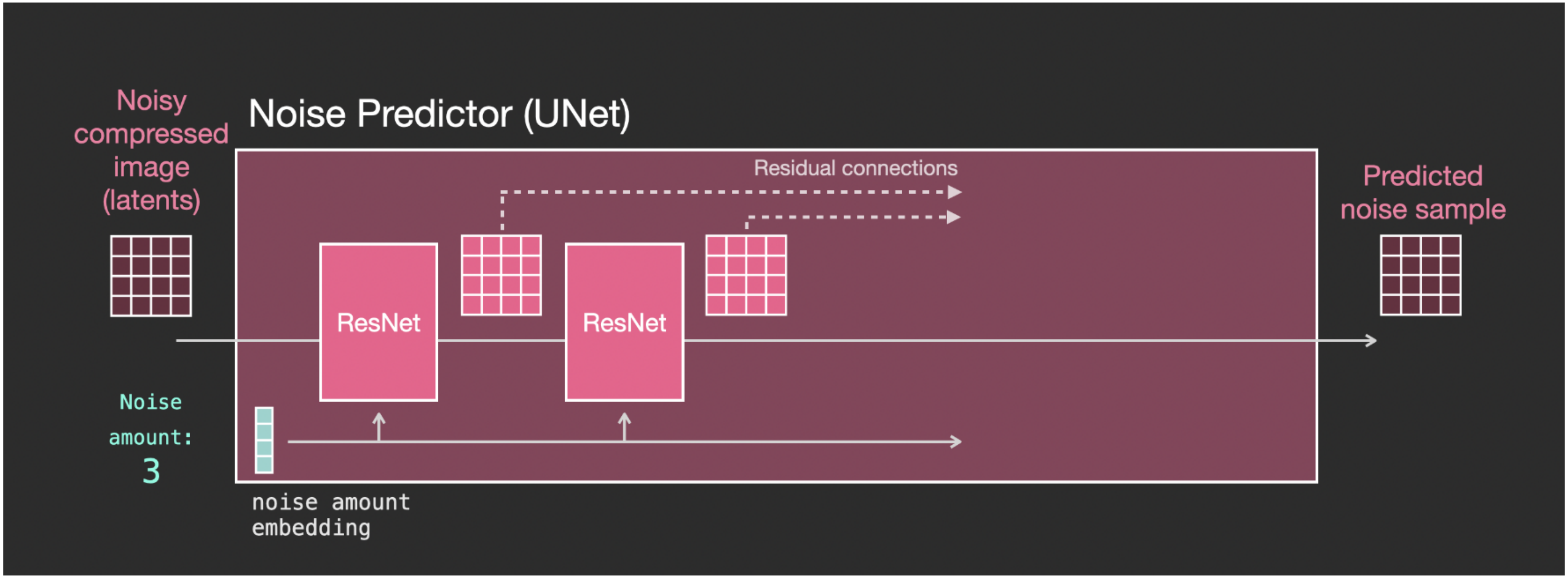


Unet Predictor (Without Text)



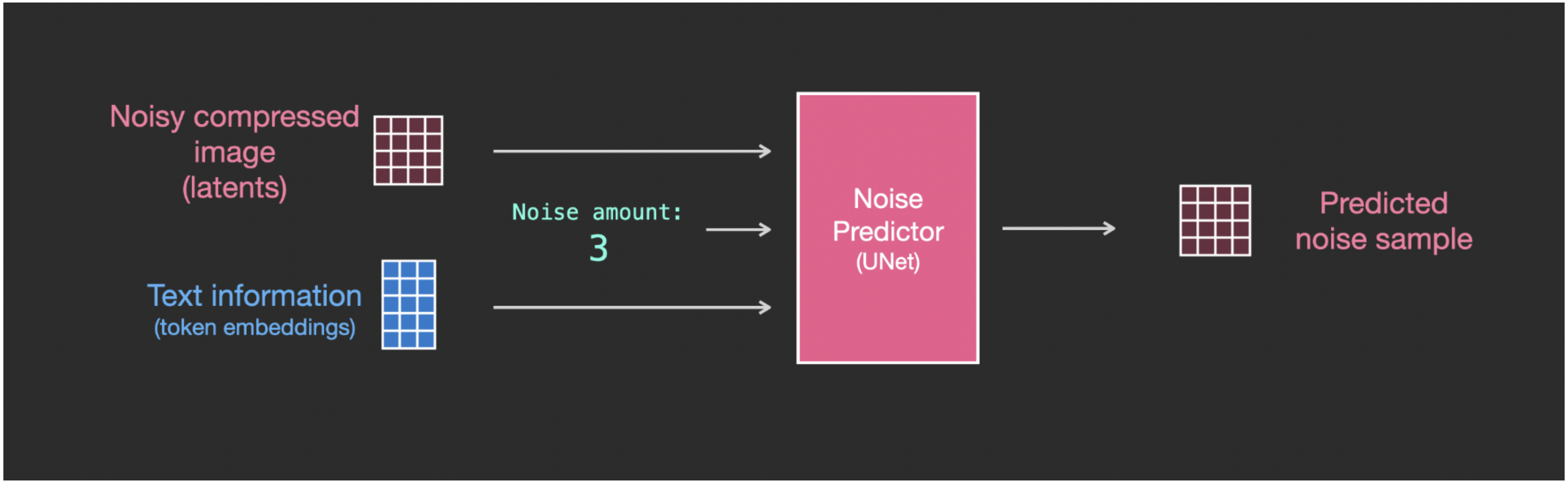
Reference: The Illustrated Stable Diffusion

Unet Predictor (Without Text)



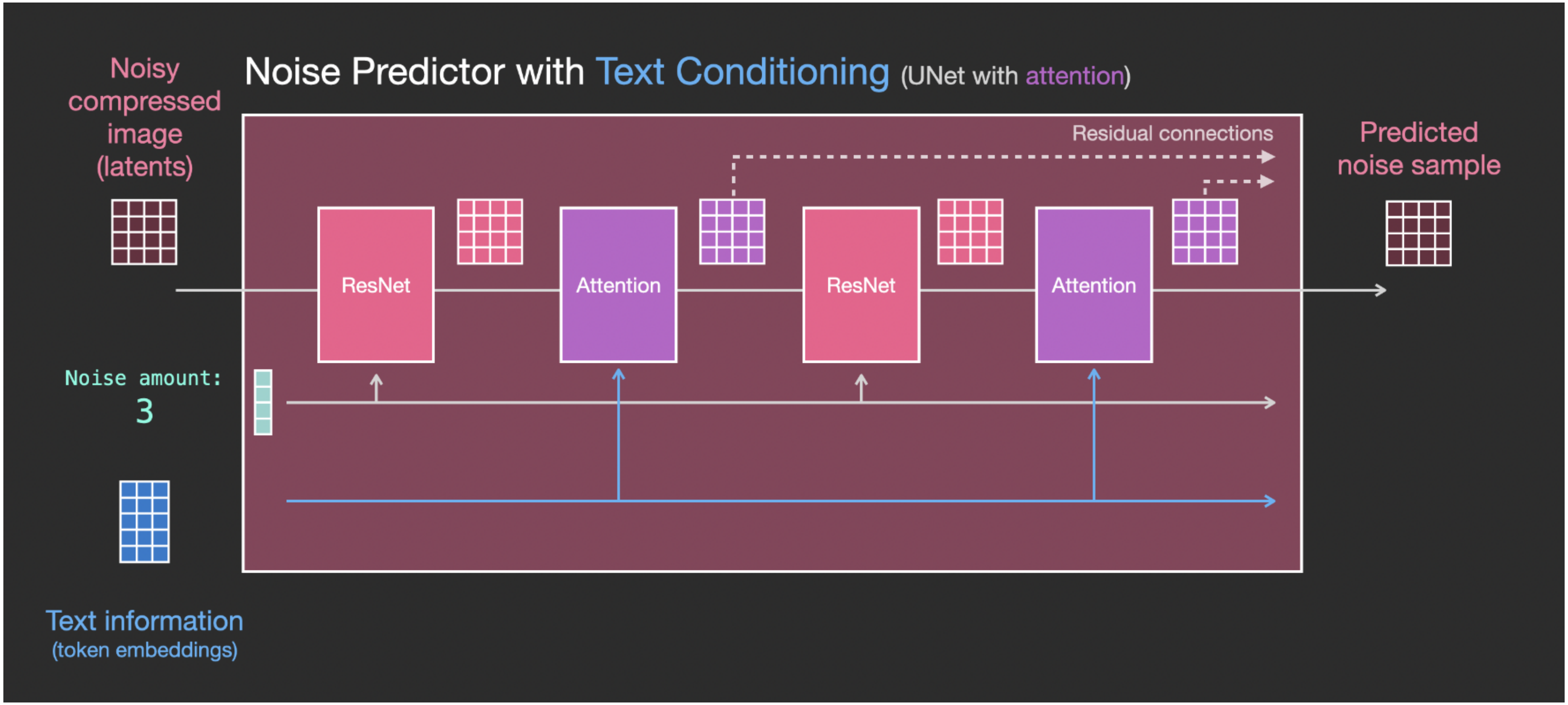
Reference: The Illustrated Stable Diffusion

Unet Predictor (With Text)



Reference: The Illustrated Stable Diffusion

Unet Predictor (With Text)



Reference: The Illustrated Stable Diffusion