Basic of Diffusion Models

Recap

Why study images?

Basic of pixels.

Basics of RGB.

Basics of channels.

What is noise?

Image representation in Python: pillow

Python libraries for Images

Agenda

Image representation

Latent space

Encoder-decoder architecture

Autoencoder

Variational autoencoder [VAEs]

Generative Adversarial Networks [GANs]

Diffusion Models

1. Image Representation

Represent

$$\begin{bmatrix} a_{00} & a_{01} & a_{02} & a_{03} & a_{04} & a_{05} & a_{06} & a_{07} \\ a_{10} & a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} & a_{17} \\ a_{20} & a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & a_{26} & a_{27} \\ a_{30} & a_{31} & a_{32} & a_{33} & a_{34} & a_{35} & a_{36} & a_{37} \\ a_{40} & a_{41} & a_{42} & a_{43} & a_{44} & a_{45} & a_{46} & a_{47} \\ a_{50} & a_{51} & a_{52} & a_{53} & a_{54} & a_{55} & a_{56} & a_{57} \\ a_{60} & a_{61} & a_{62} & a_{63} & a_{64} & a_{65} & a_{66} & a_{67} \\ a_{70} & a_{71} & a_{72} & a_{73} & a_{74} & a_{75} & a_{76} & a_{77} \\ a_{80} & a_{81} & a_{82} & a_{83} & a_{84} & a_{85} & a_{86} & a_{87} \\ a_{90} & a_{91} & a_{92} & a_{93} & a_{94} & a_{95} & a_{96} & a_{97} \end{bmatrix}$$

We can represent visual image in high dimensions matrix. High dimension leads to more computing power.

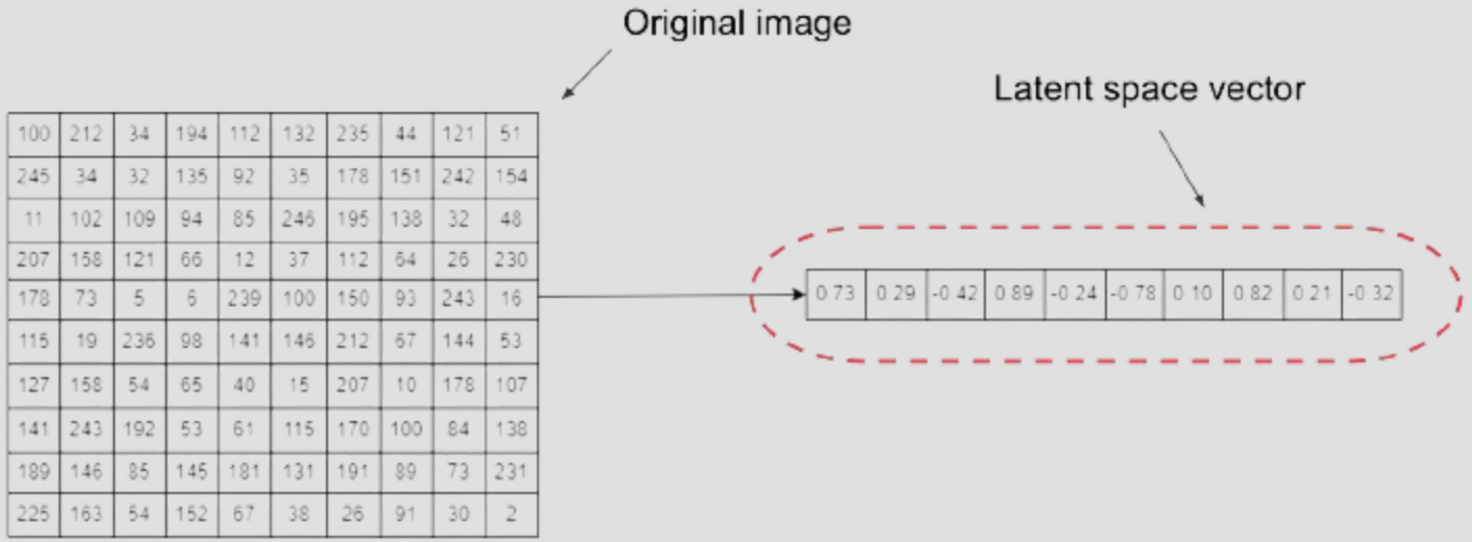
2. Latent Space

Latent space is a lower-dimensional representation of data called dimensionality reduction.

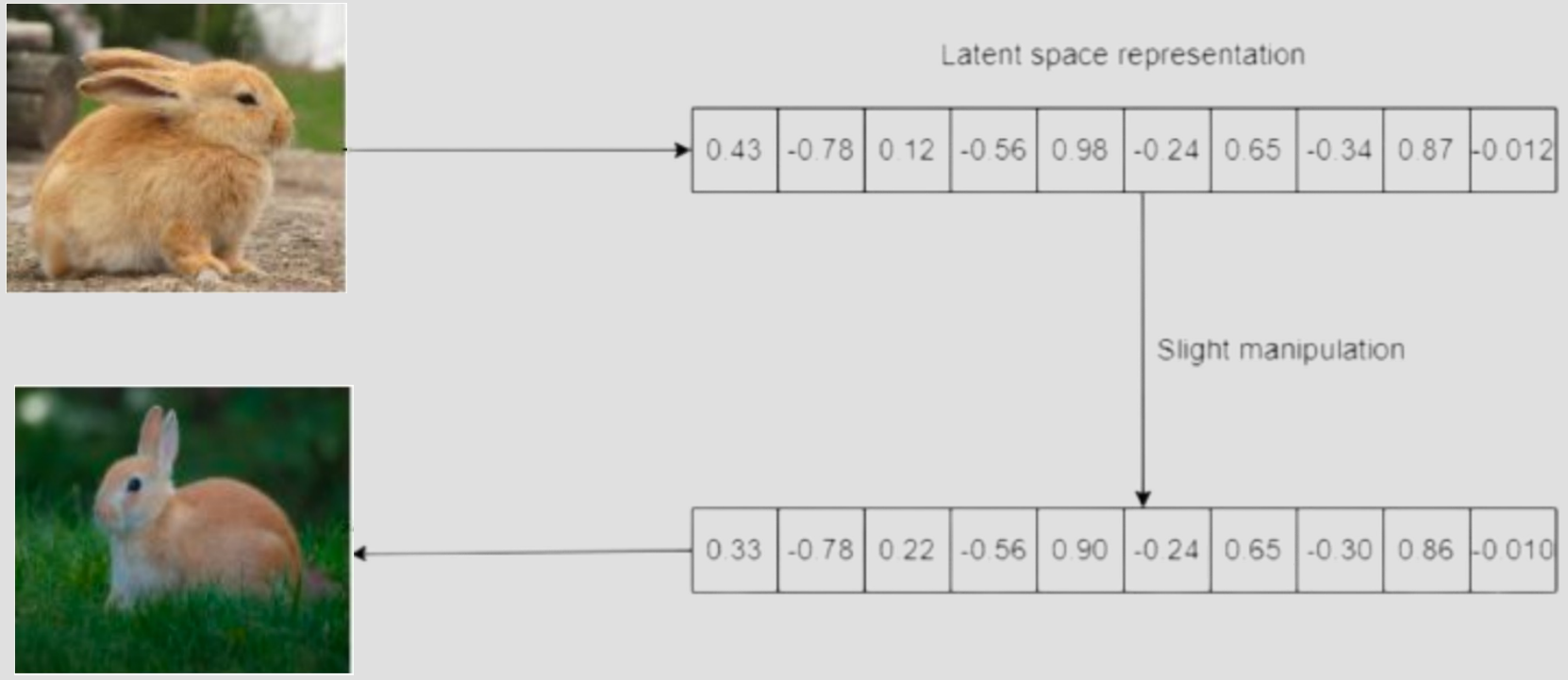
Captures essential features or patterns of the higher-dimensional data.

By manipulating these latent space vectors, we can generate new images that share similarities with the original dataset.

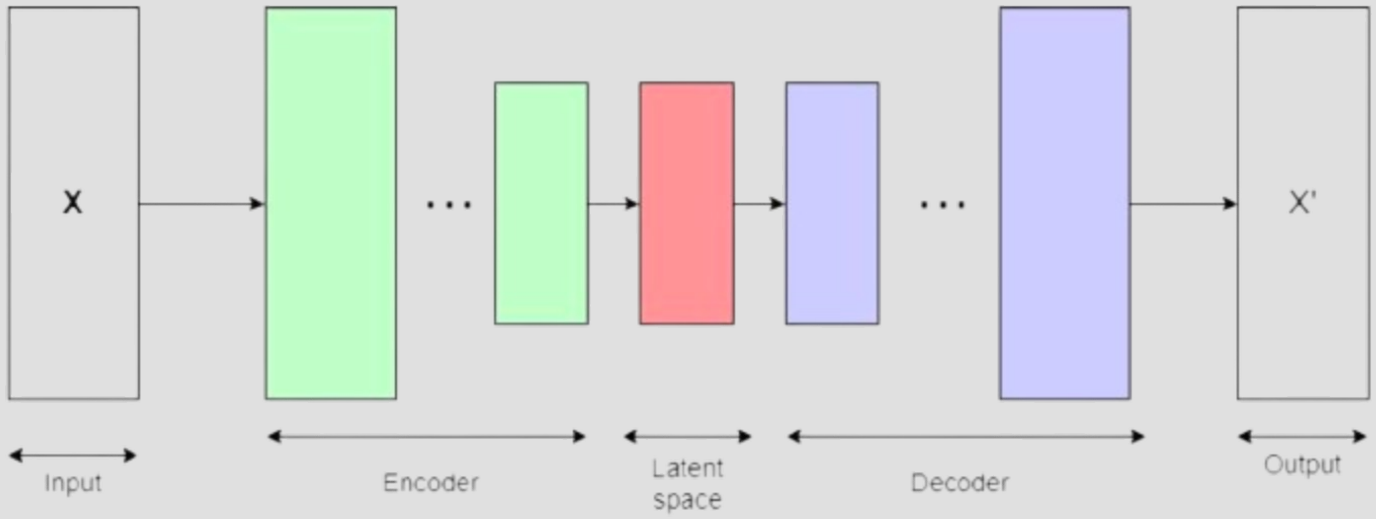
2. Latent Space



2. Latent Space



3. Encoder Decoder Architecture



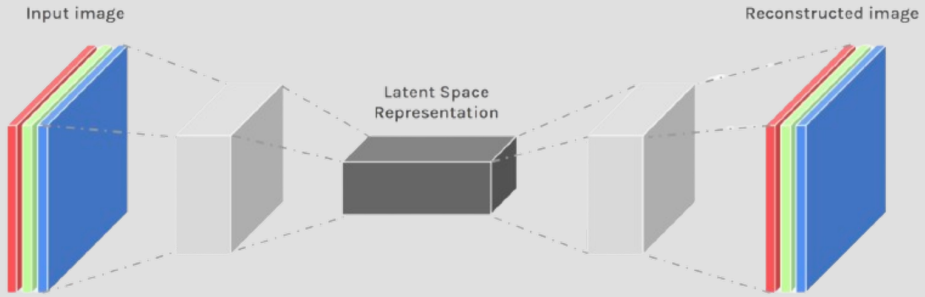
3. Encoder Decoder Architecture

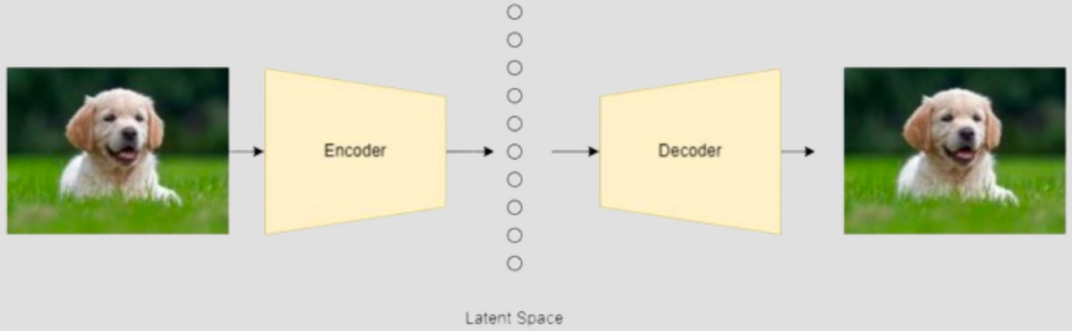
Encoder: One or more layers of neurons that convert the input into an encoded representation.

Latent Space: Lower dimensional representation of the input that captures the most salient features.

Decoder: One or more layers of neurons that recreate the image from the latent space.

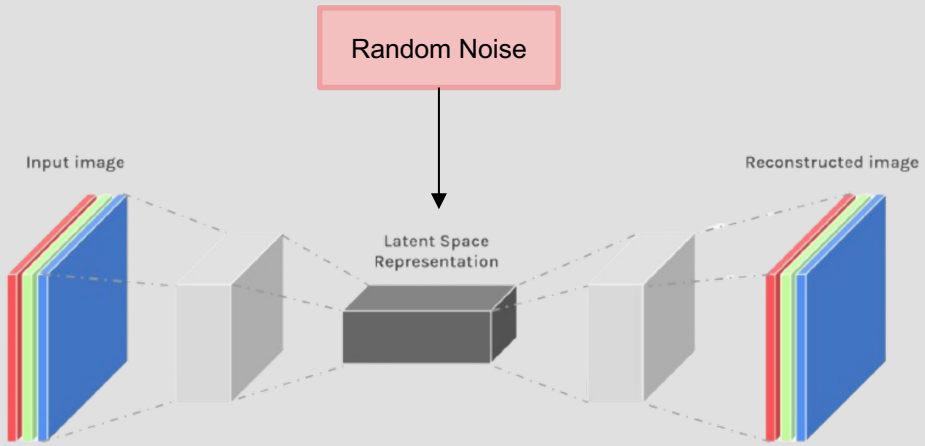
4. Autoencoder

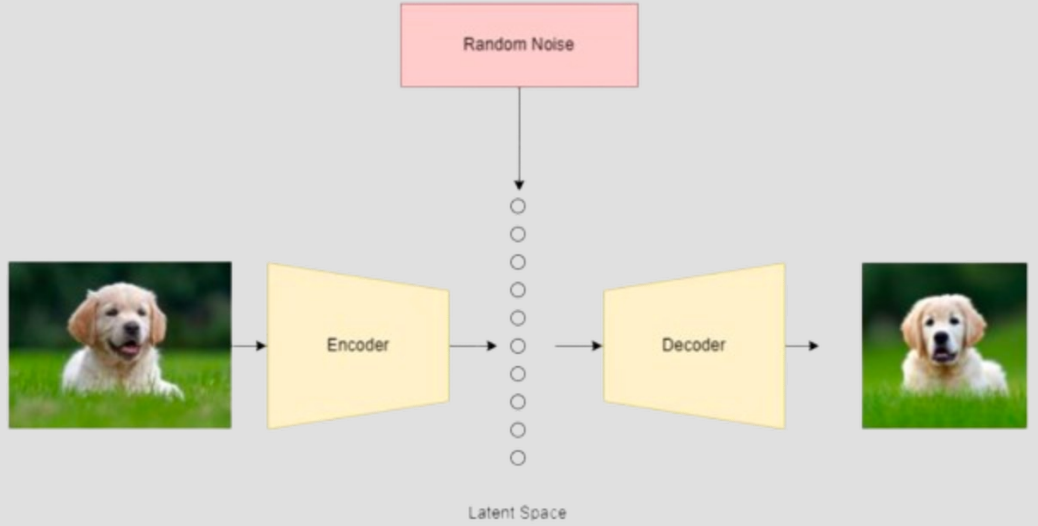




Autoencoders are neural networks that use encoder-decoder architecture to create same images.

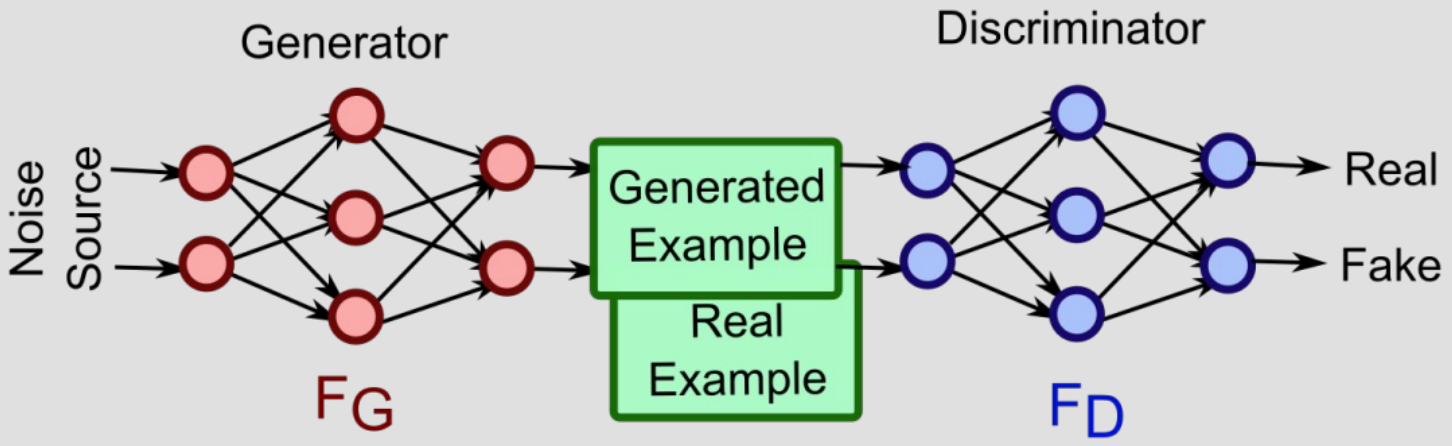
5. Variational Autoencoder





Variational autoencoders are neural networks that use encoder-decoder architecture to create similar image by adding noise.

6. Generative Adversarial Networks[GANs]

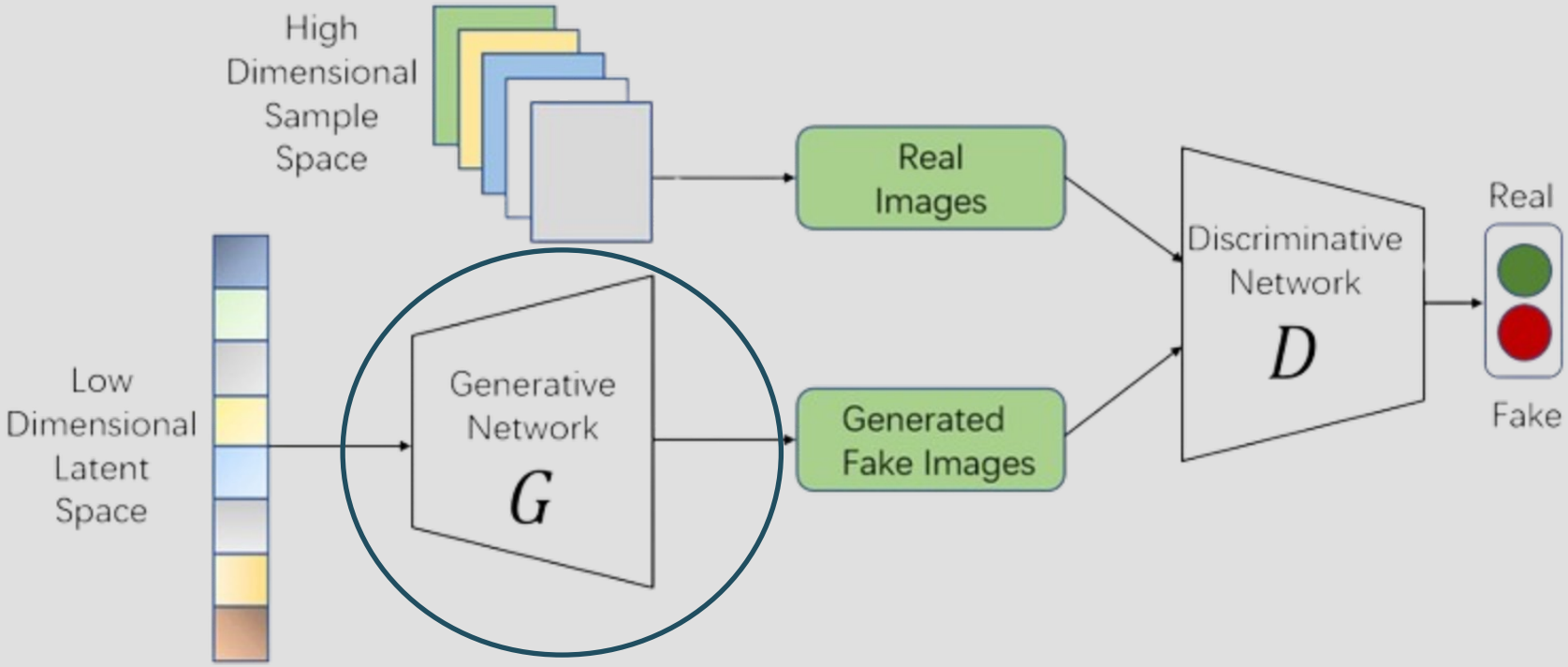


Consists of two networks: **Generator** and **discriminator.**

Generator creates new images.

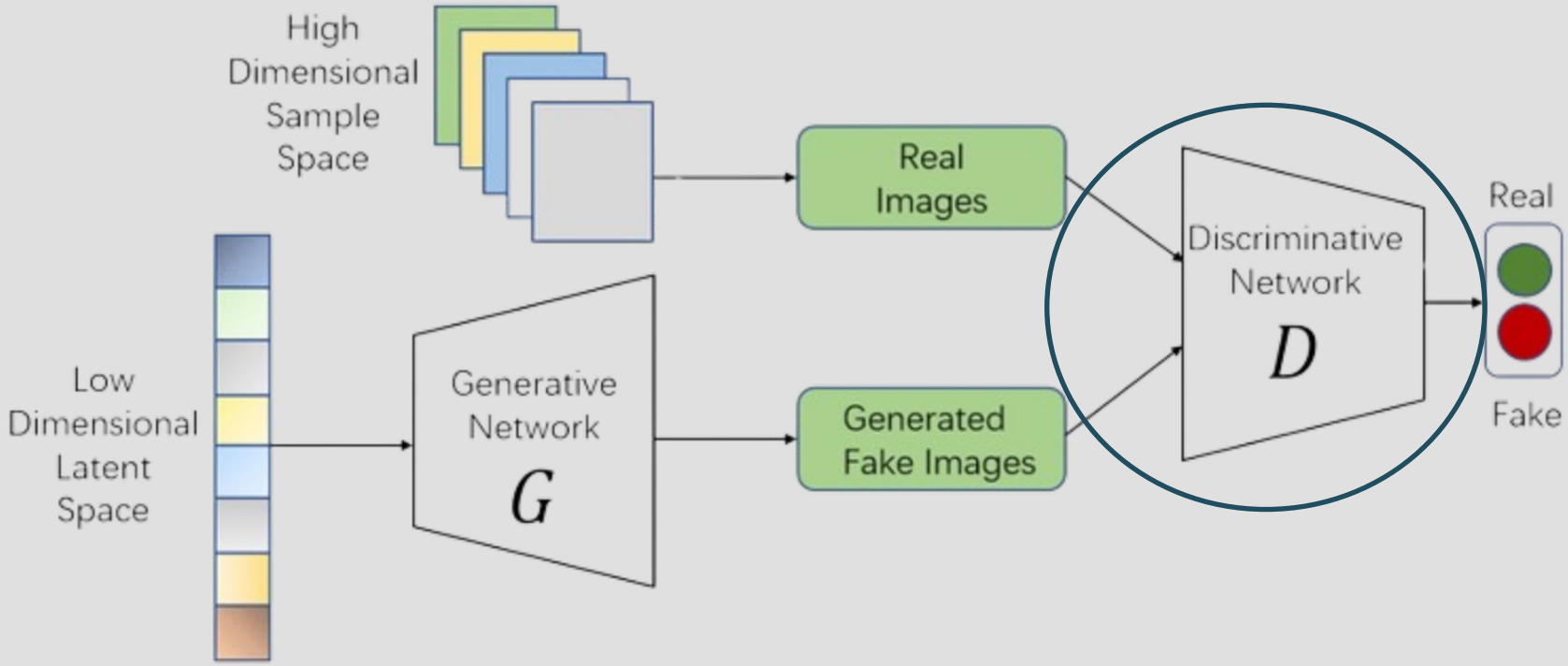
Discriminator tries to determine whether new images are real or fake.

6. GANs -> Generator



Generator: Creates fake data that looks as realistic as possible

6. GANs -> Discriminator



Discriminator: Tells the difference between real and fake data

6. GANs -> Adversarial

The two networks (Generator and discriminator) try to one up each other.

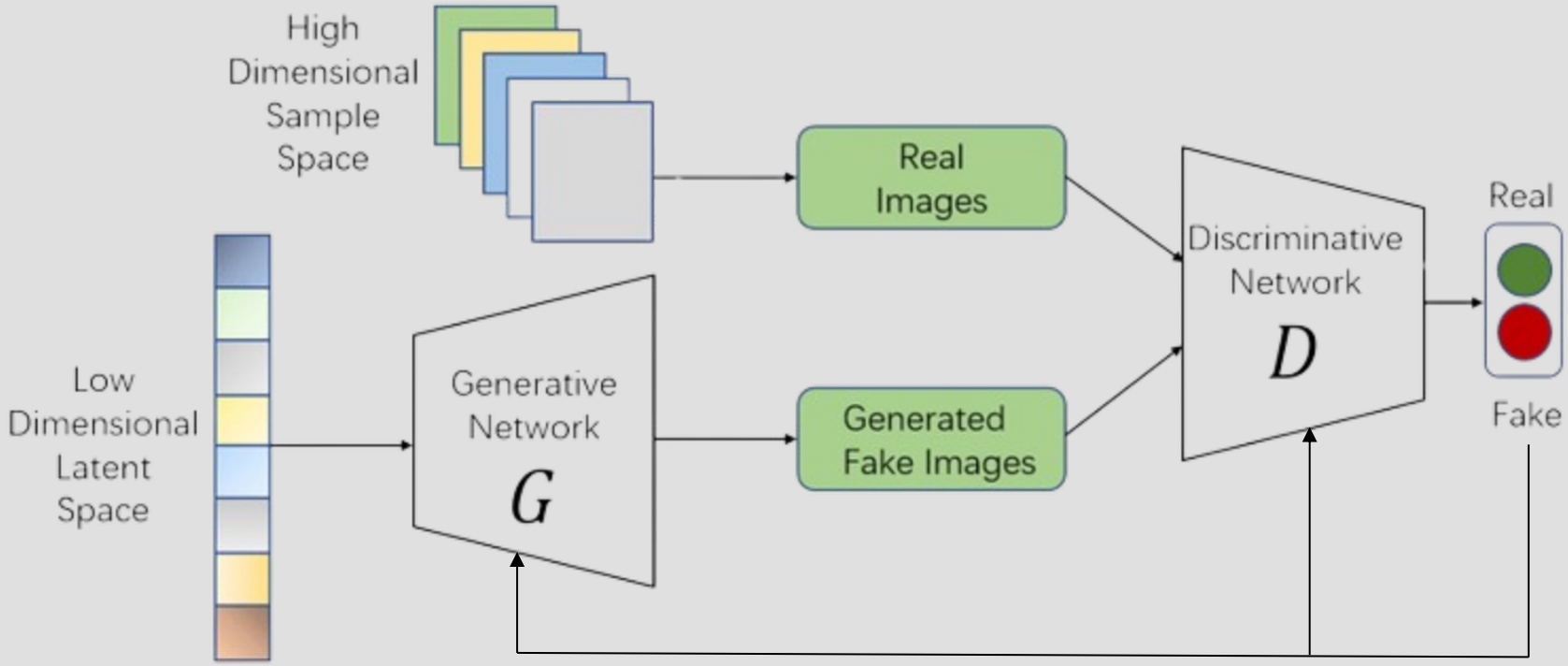
The generator tries to generate images that are closer and closer to the real images in the data set .

The discriminator tries to determine whether the generated images lie in the distribution described by the data set .

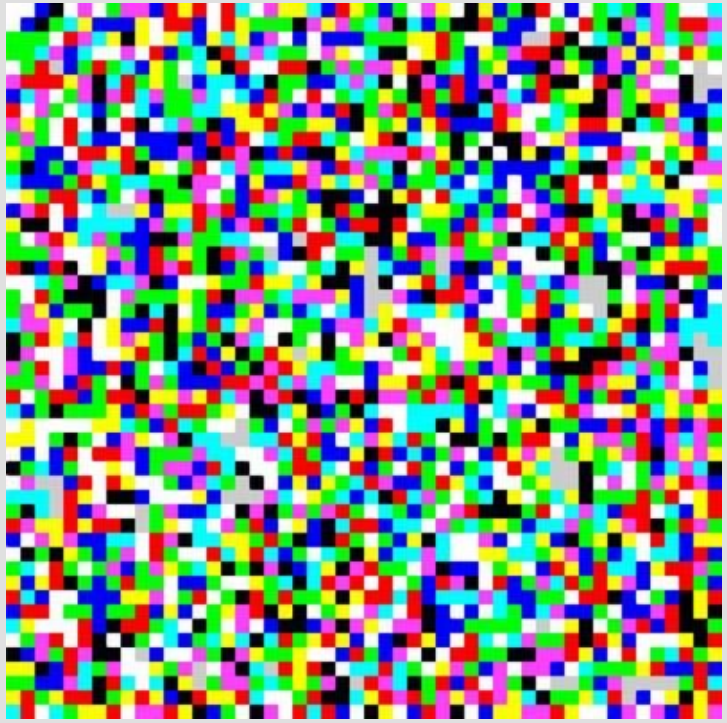
The complete network is trained as a whole .

After training, the discriminator is discarded.

6. GANs -> Training

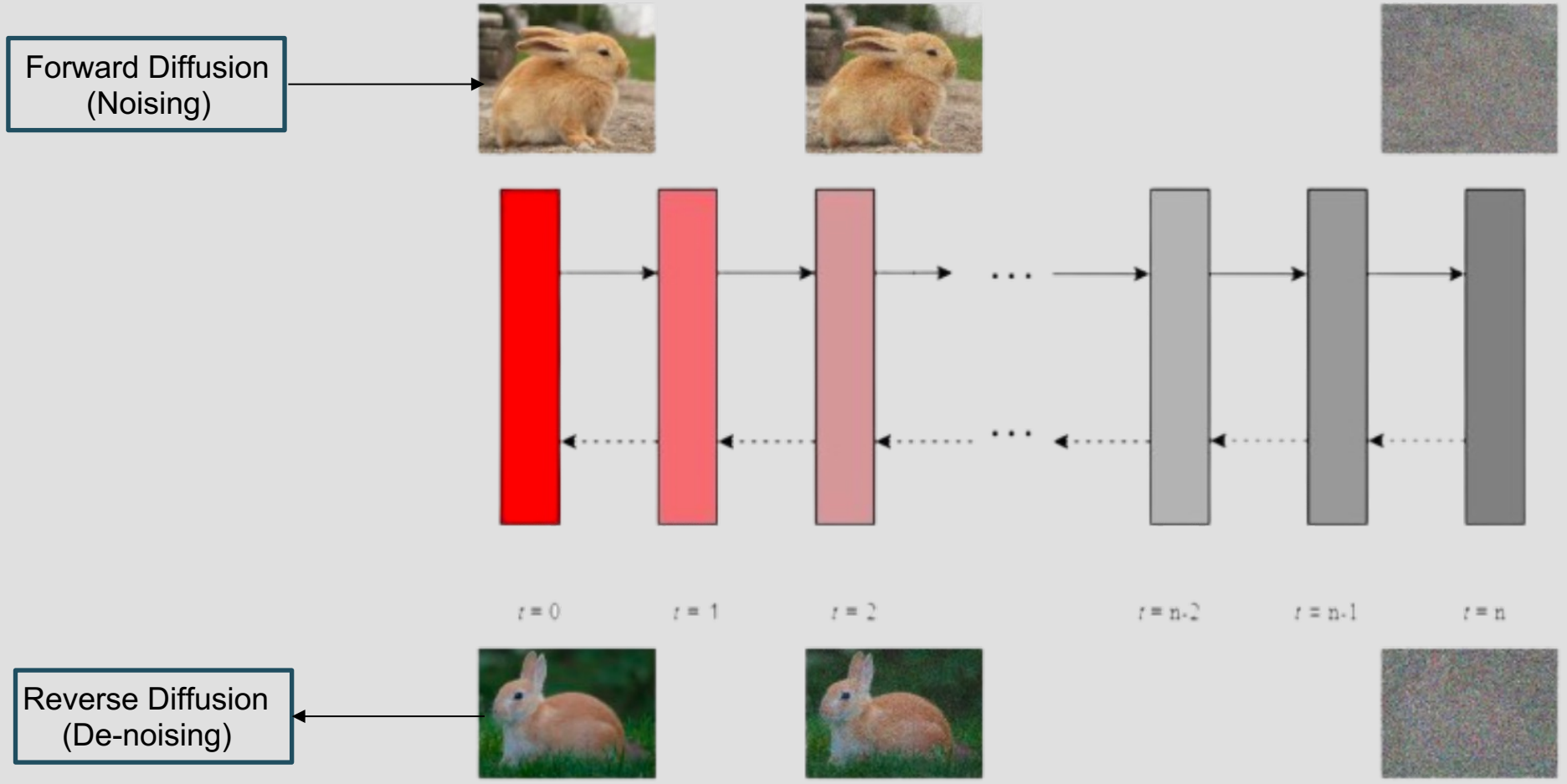


7. Diffusion Models



Diffusion models generate images by moving from noise to image

7. Forward and Reverse Diffusion



7. Forward and Reverse Diffusion

Happens during **training**.

Add Gaussian noise to the data at each step.

The rate at which noise is added is determined by a scheduler .

β controls the amount of noise added at each step.

The point is to enable the network to predict the noise added at any step

Happens during **prediction**.

We start with pure noise at t = n .

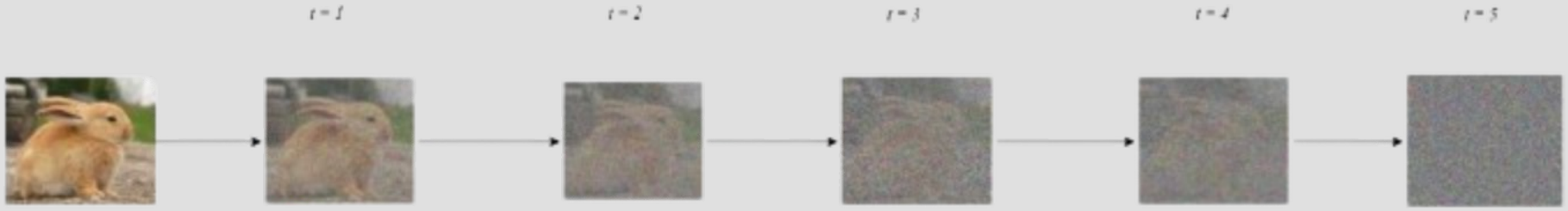
We are trying to get xt-1 from any xt .

The noise is removed by the scheduler which is controlled by β .

Reverse diffusion can be performed in fewer timesteps as compared to the number of timesteps used to train the model.

7. Diffusion Process

Images are used as training data to train the model to predict noise in the forward diffusion process



A random noisy patch is then denoised in the reverse diffusion process to produce a new image

7. Noise Predictor: U-Net

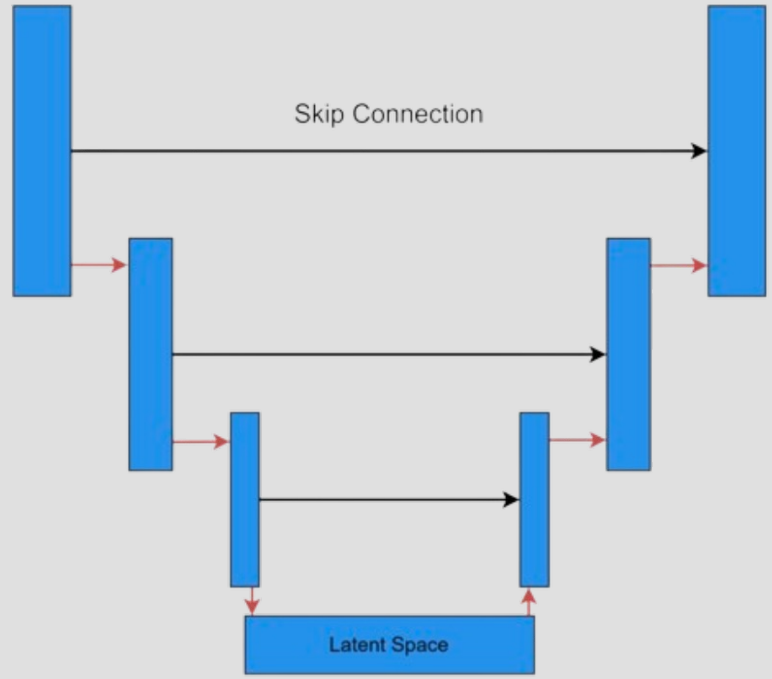
A U-net is used to predict noise in the diffusion model.

It has an encoder-decoder architecture.

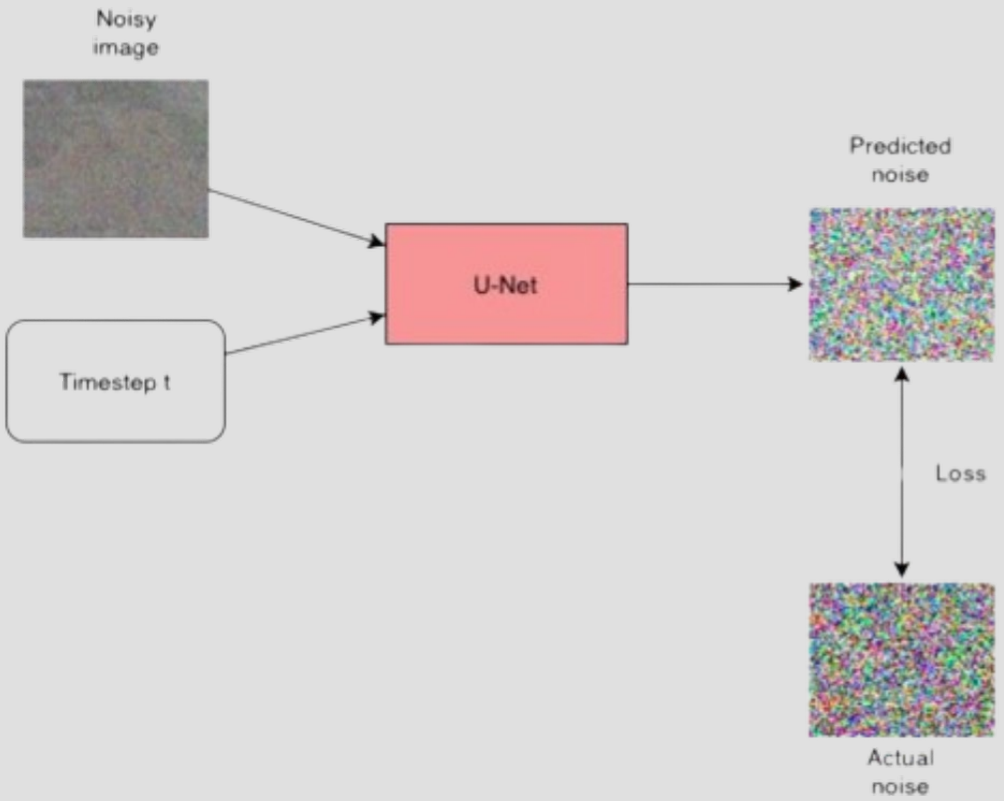
Encoder (left): Converts the input image into a lower-dimensional latent space and extracts meaningful features at multiple stages.

Decoder (right): Gradually increasing dimensions to produce a final output with the same size as the input image.

Skip connections are uses to pass information from the encoder to the decoder .



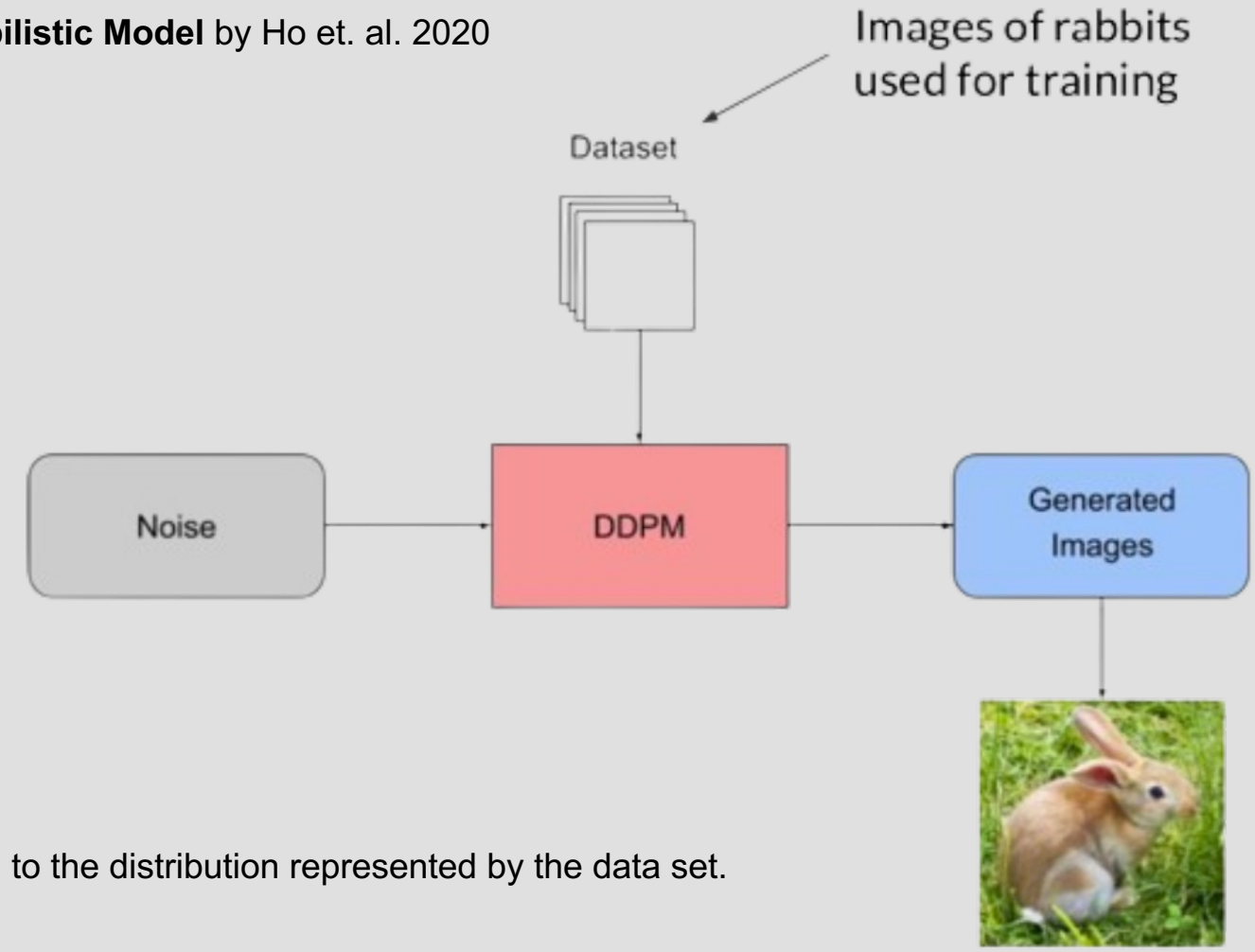
7. Noise Predictor: U-Net



We train the network to predict the noise given an image and a timestep.

7. Diffusion Model: Unconditional

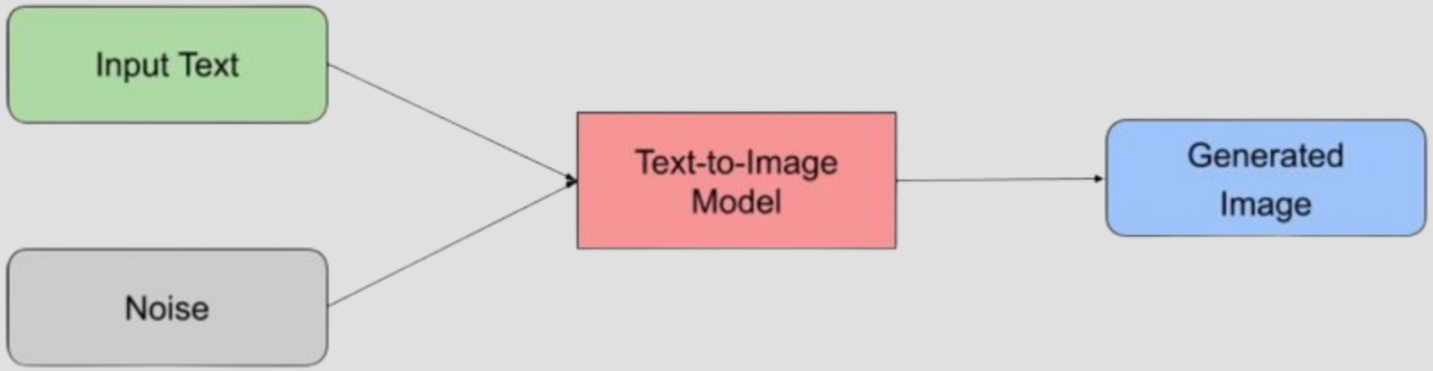
Denoising Diffusion Probabilistic Model by Ho et. al. 2020



- Noise is the only input.
- Generated images belong to the distribution represented by the data set.

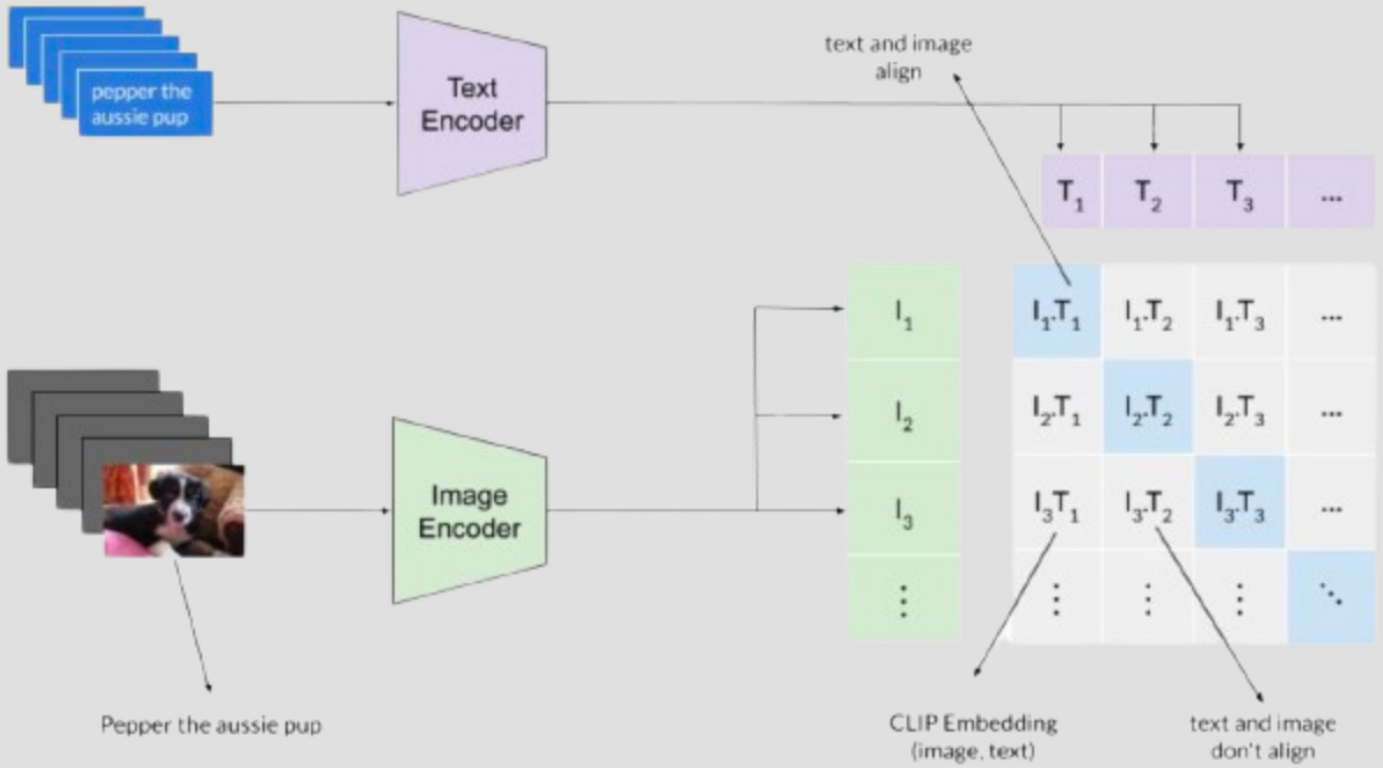
7. Diffusion Model: Conditional[T2I]

High-Resolution Image Synthesis with Latent Diffusion Models by Rombach et. al.



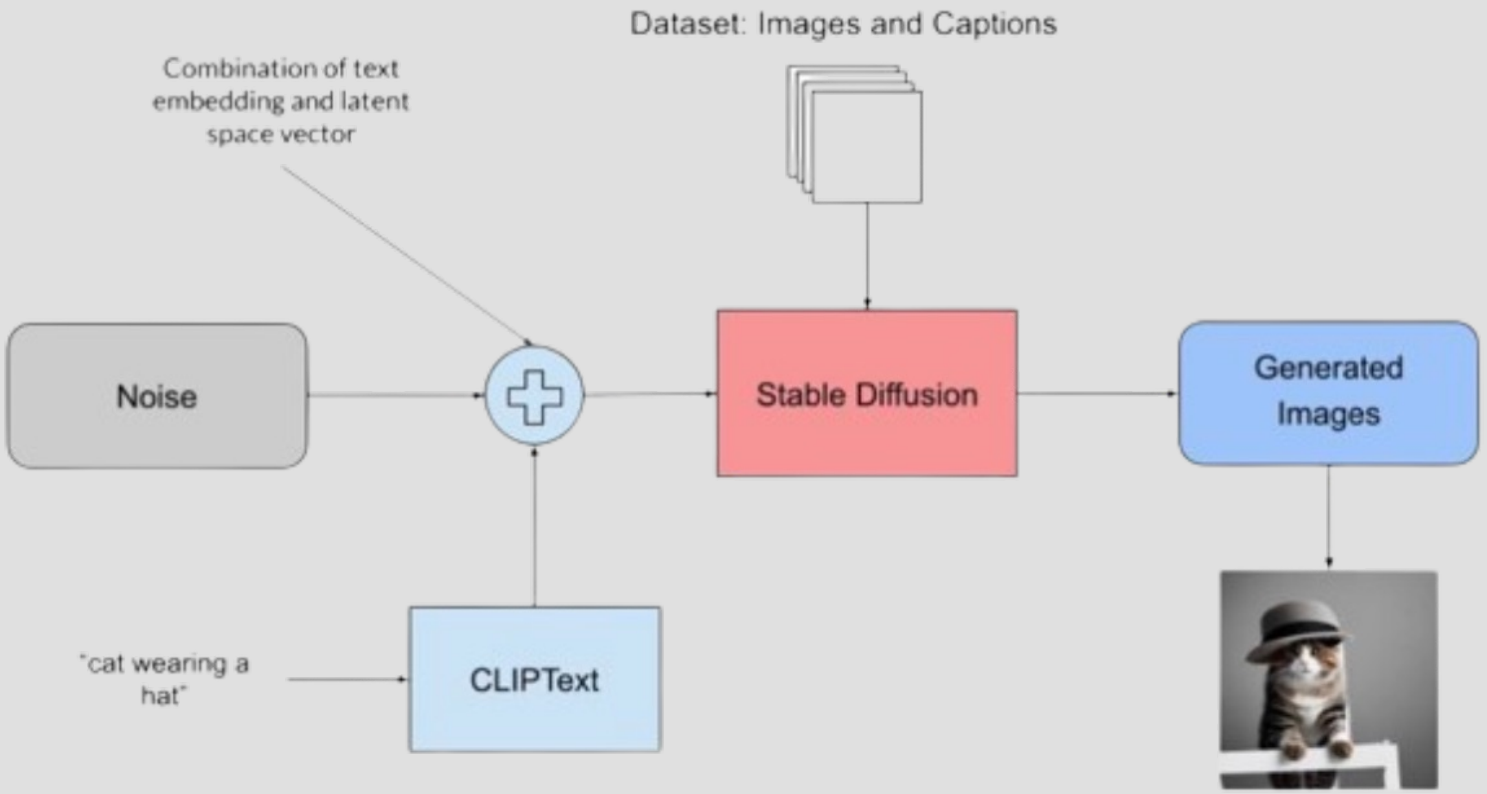
- Stable Diffusion achieves this, along with a few other optimizations.

7. CLIP Embedding



- We transform text and images into their respective embedding vectors using the CLIP model.
- These embeddings guide the denoising process

7. Stable Diffusion



Thank You !