Deep Learning
Lecture 11

Generative Models

Prof. Dr. Rainer Herrler

Phone: 09721/ 940-8710

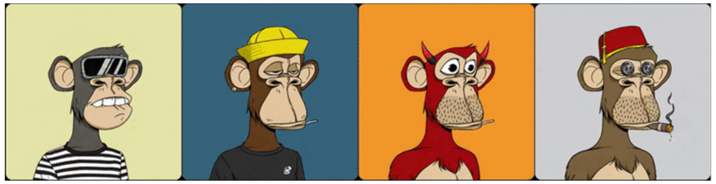
Email: rainer.herrler@fhws.de

Pictures from Wikipedia / Pixabay

Some Pictures generated with Stable Diffusion

What are Generative AI Models

- Generative Models create new data in accordance to the underlying distribution of the training samples.
- Examples
  - ThisPersonDoesNotExist
  - ThisApeDoesNotExist

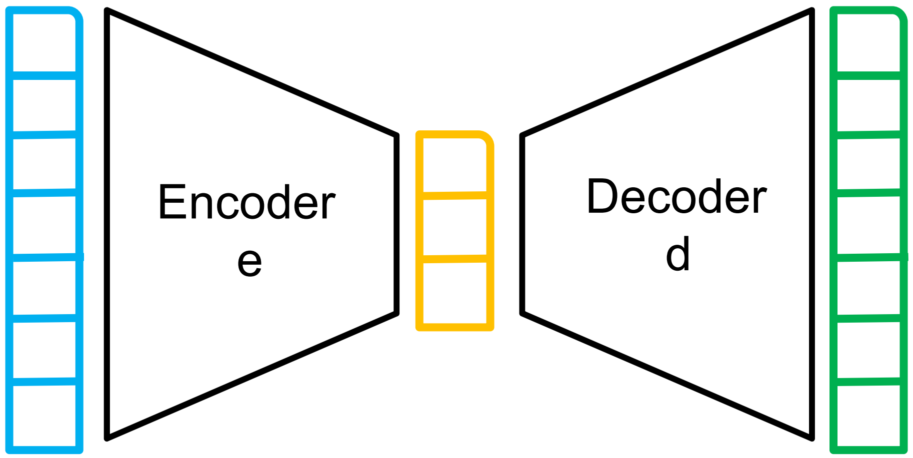


- Technique Overview
  - Autoencoder
  - Variational Autoencoder
  - Generative Adversarial Networks
  - Diffusion Models

Autoencoder

What is an autoencoder ?

- Unsupervised learning Task using a Neural Network with a bottleneck
- No Labels required, the loss function checks the recreation of the original sample



Input vector x from $\mathbb{R}^n$

Encoded value in $\mathbb{R}^l$ (Latent Space)

Decoded value in $\mathbb{R}^n$

Could such an network be useful?

Theoretical applications of autoencoders

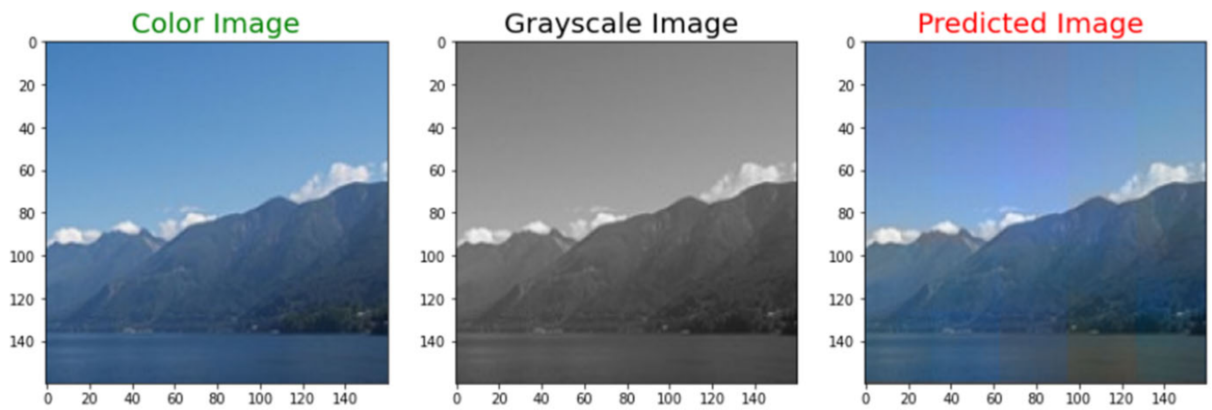
- Compression of data
  - Alternative to PCA
  - Input sample x is identical to label during training
  - The **latent vector** contains a compressed version of the original image
  - Once the network is trained the decoder can be used to reconstruct images from the latent representation.

- Denoising autoencoder
  - Noise is added to the input sample during training, but they are checked against the noiseless original
  - The network will learn to reconstruct the original image

Not really autoencoders but very related tasks

In following examples we have the encoder task given and just the decoder is learned.

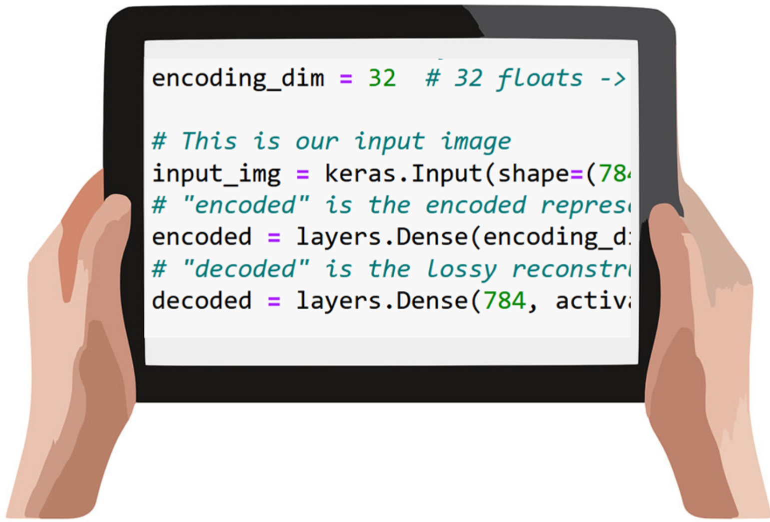
- Image upscaling
  - We use down-sampled images as input and original images as label
  - the decoder will be trained to upscale the image
- Image colorization
  - We remove color information from images to learn the decoder

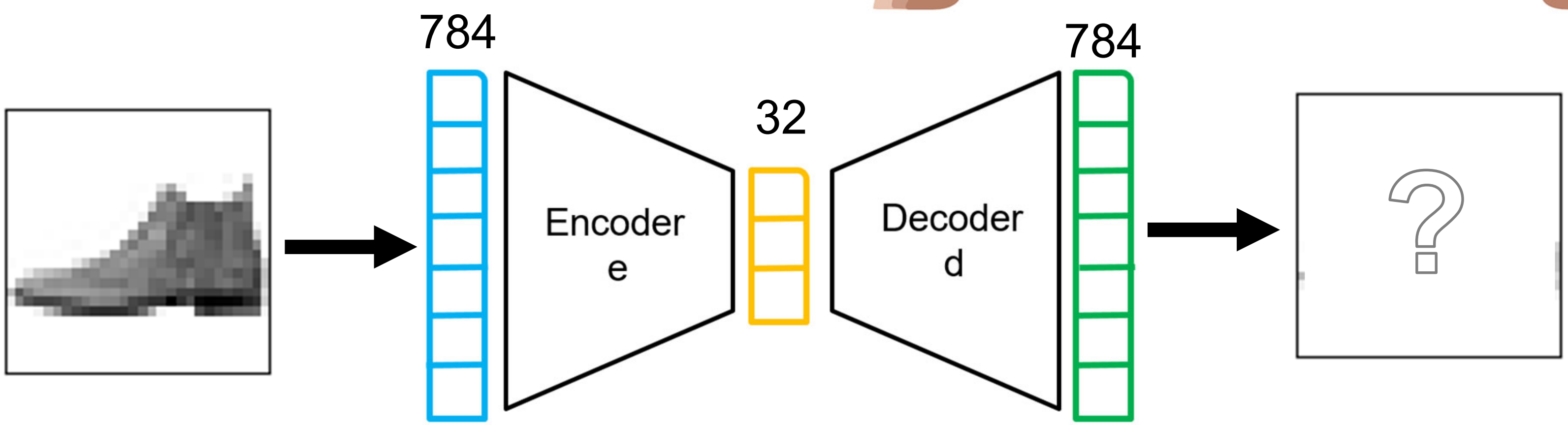


https://www.kaggle.com/code/theblackmamba31/autoencoder-grayscale-to-color-image

A Simple Autoencoder

[DL_030_SimpleAutoEncoder.ipynb]()
[DL_031_AutoEncoderCNN]()
[DL_032_AutoEncoderForDenoising]()





What did we learn from the Simple Autoencoder

- We were able to store the images of 28x28 Pixels in a latent space of 32 Neurons
- However the latent space is not structured
  - Random values have no meaningful result
  - Even small changes in the latent space can have big impact
  - Same similar pictures are not necessarily close in the latent space

- Loss function can be MSE or Binary Crossentropy

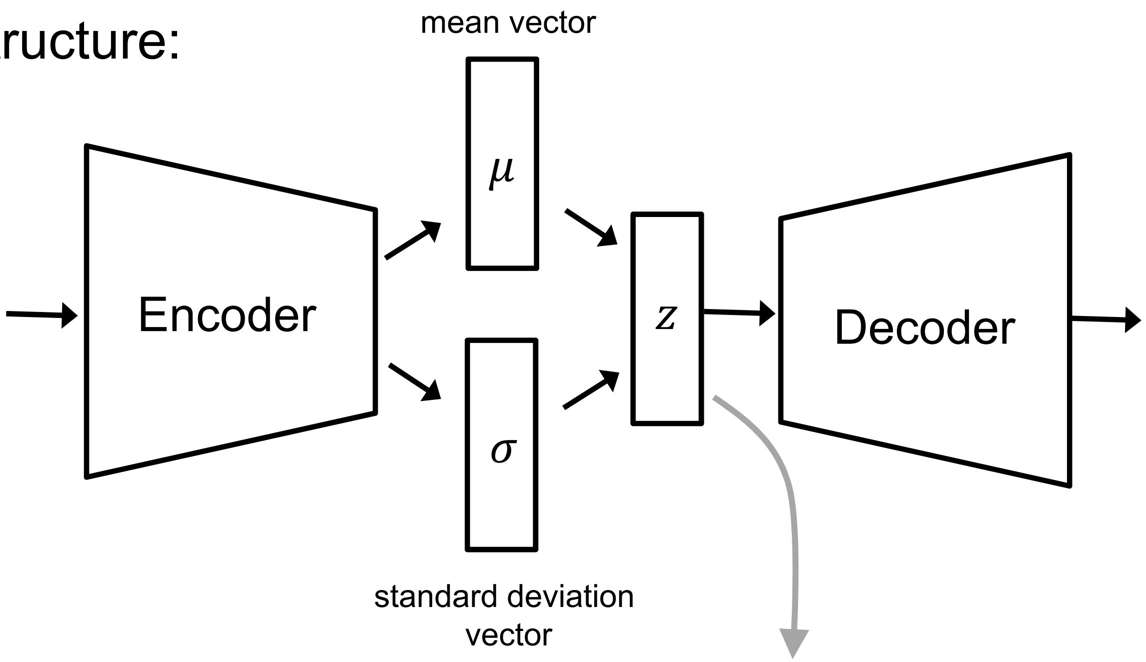
Summary on basic autoencoders

- Autoencoders are a unsupervised learning Task, no Labels required
- The Autoencoder consists of
  - An Encoder to maps input to a lower dimensional representation
  - The latent space (Representation between the two encoders)
  - The Decoder learns the inverse translation
- Theoretical Applications of autoencoders
  - Denoising data
  - Compression of data
  - Various other: Upscaling images, Colorizing…
- Importance
  - Alternative to PCA or other dimensionalty reduction techniques we know of from last year
  - Not state of the art for Compression and Upscaling
  - Base idea for further advanced algorithms
- Article of François Chollet
  - https://blog.keras.io/building-autoencoders-in-keras.html

Variational Autoencoder
VAE

What is an variational autoencoder?

Structure:



$$z = \mu + \sigma \odot \varepsilon$$

- Very similar to an autoencoder
- Additional structures are introduced ro “regularize” the encoding distribution

Applications of variational autoencoders

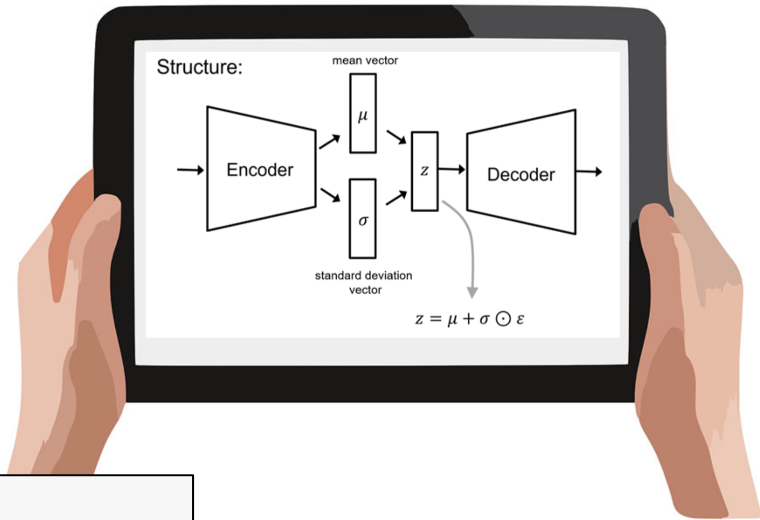
- Drawbacks of pure autoencoders:
  - A pure autoencoder has no incentive to organize the latent space well,
  - the only goal of a pure autoencoder is to minimize the reconstruction loss
- Idea:
  - Instead of encoding an input to a singe point, we encode is as a distribution around a single point in latent space
  - A distribution can be described as a mean value and a standard deviation.
  - During training, we sample from the encoded distribution to decode the
  - Effects:
    - Close Points in latent space lead to similar results
    - Latent space is better structured
- Articles:
  - https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73
  - https://towardsdatascience.com/intuitively-understanding-variational-autoencoders-1bfe67eb5daf

Applications of variational autoencoders

- Representation Learning
  - Features of the latent space can be related with “recognizeable features” in the image.
  - A latent representation of person faces can have features like haircolor, smile etc, usually the emergence of these features can not be controlled
  - Contrastive learning
- Data Generation
  - You can create meaningful new data from randomly creating a new latent space sample
- Morphing Images
  - If you have two images generated from samples in the latent space you can generate morphs by interpolating a path in the latent space

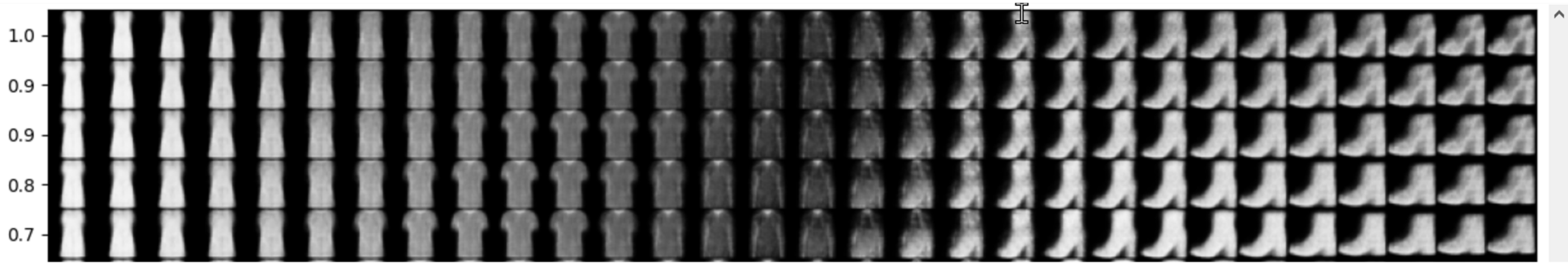
A Variational Autoencoder

DL_030_SimpleAutoEncoder.ipynb



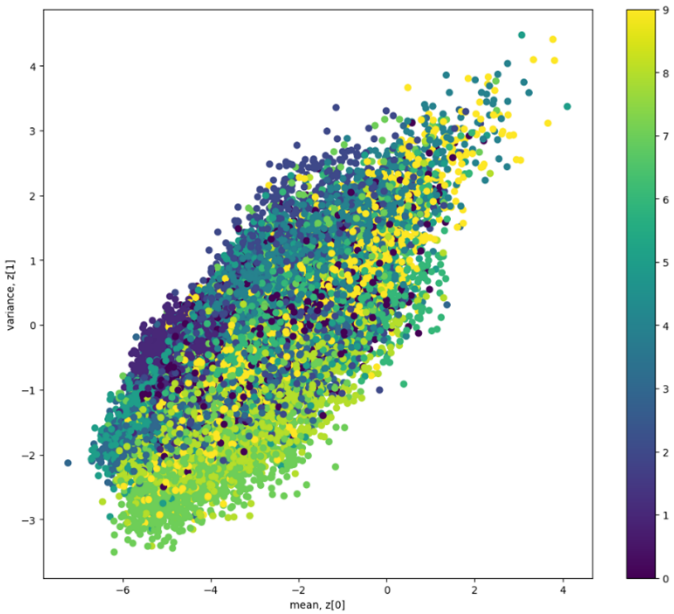

```
# Build the encoder
def encode(self, data):
    x = self.encoder(data)
    z_mean, z_log_var = self.z_mean(x), self.z_log_var(x)
    return z_mean, z_log_var

# Build the reparameterization/sampling layer
def reparameterization(self, z_mean, z_log_var):
    batch = tf.shape(z_mean)[0]
    dim = tf.shape(z_mean)[1]
    epsilon = tf.keras.backend.random_normal(shape=(batch, dim))
    z =  z_mean + tf.exp(0.5 * z_log_var) * epsilon
    return z
```



Observations:
- The latent space contains more valid examples
- Similar latent vectors lead to similar generated images
- We can see transitions from one class to another

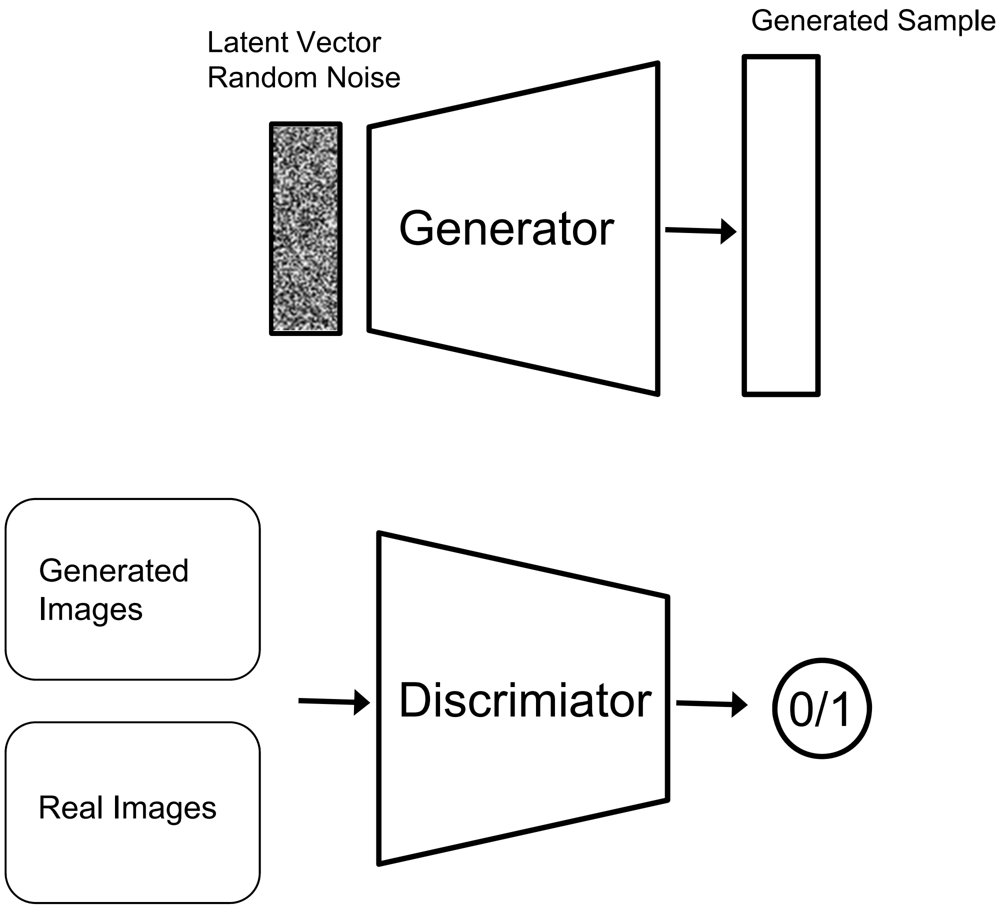
Comments
- Here we have labels very often we autoencode things of the same class that vary just in details



Each color represents one of the 10 classes
Each point is the latent representation of one original image.

Generative Adverserial Networks

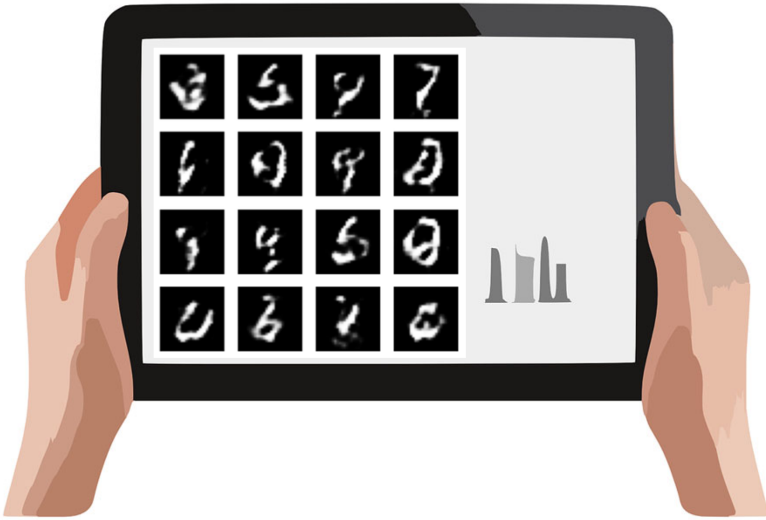
Working Principle of GANs



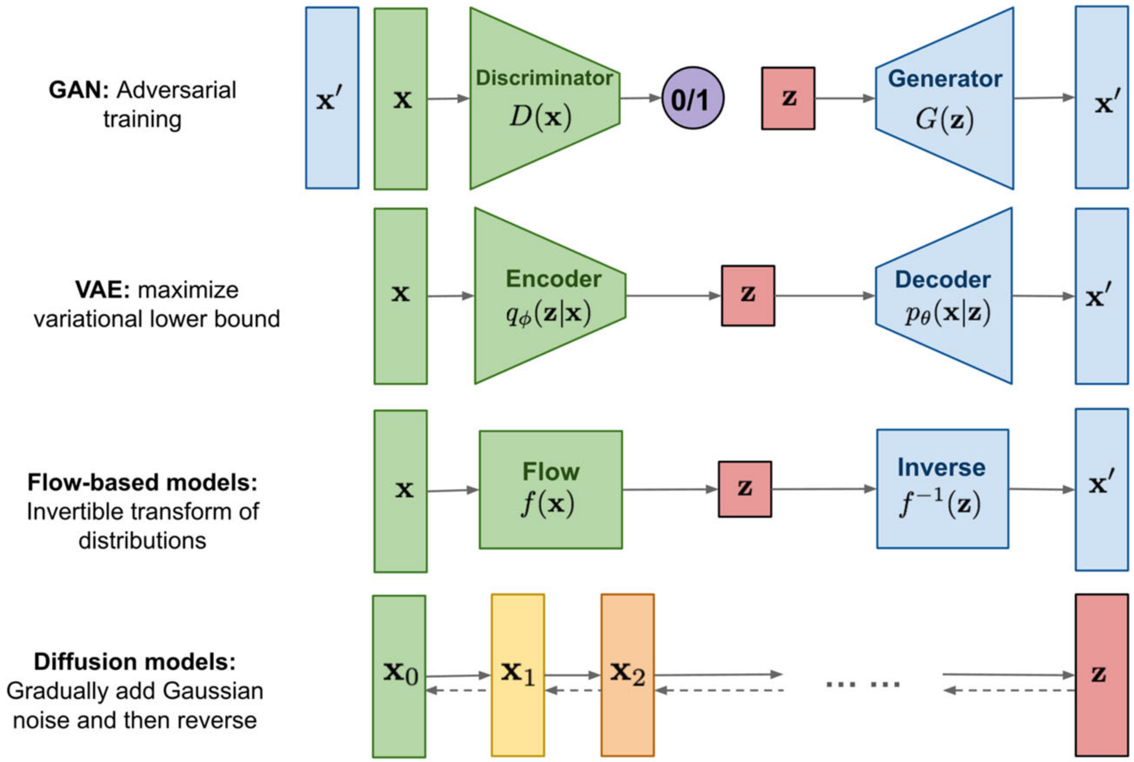
Working Principle of GANs

- Generator model
  - The generator takes a fixed length vector as input and generates a sample in the target domain
- Discriminator model
  - The discriminator takes a sample from the target domain and predicts if it is an original sample or a generated one.
- A two player adversarial game
  - The generator tries to create samples that fool the discriminator. It will be trained by the discriminator feedback to generate better samples
  - The discriminator learns to classify those samples correctly while the quality of generated pictures increases.
- DCGAN – Deep convolutional generative adversarial network

Generative Adversarial Network



https://www.tensorflow.org/tutorials/generative/dcgan



Source: https://lilianweng.github.io/posts/2021-07-11-diffusion-models/