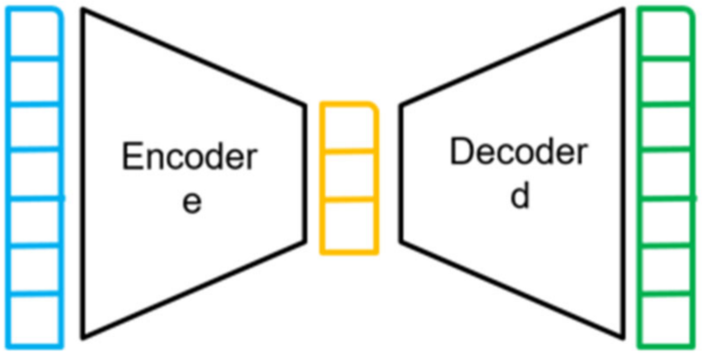
Deep Learning
Lecture 12

Generative Models Part 2
Recurrent Networks

Prof. Dr. Rainer Herrler
Phone: 09721/ 940-8710
Email: rainer.herrler@fhws.de

Pictures from Wikipedia / Pixabay
Some Pictures generated with Stable Diffusion

Recap Autoencoders

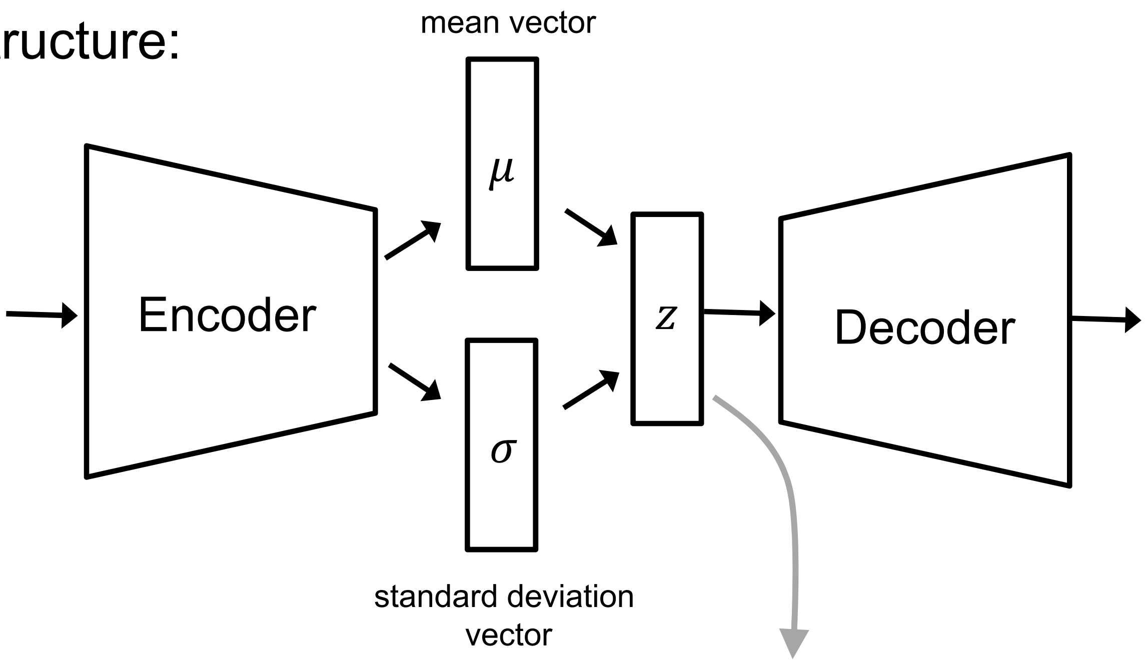


What is the autoencoder architecture ? What can it do and what cant it do ?

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

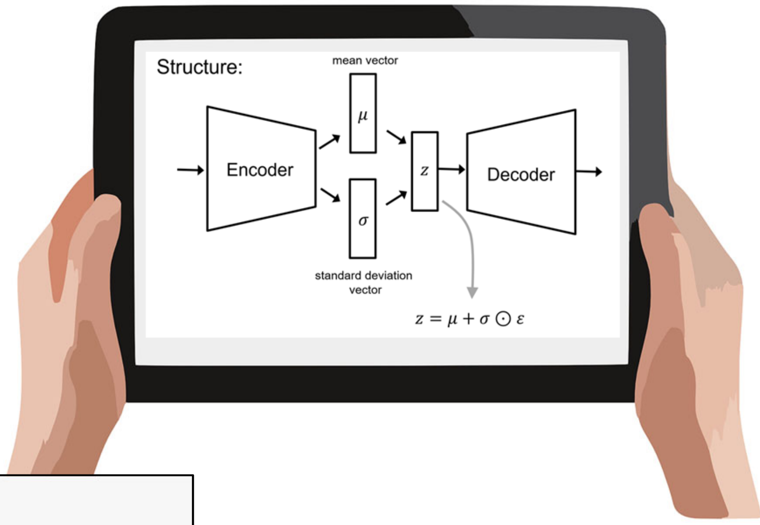
- Drawbacks of pure autoencoders:
  - A pure autoencoder has no incentive to organize the latent space well,
  - the only goal of a pure autoencoder is to minimize the reconstruction loss
- Idea:
  - Instead of encoding an input to a singe point, we encode is as a distribution around a single point in 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 it
  - 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” allows to get control over such features
- 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

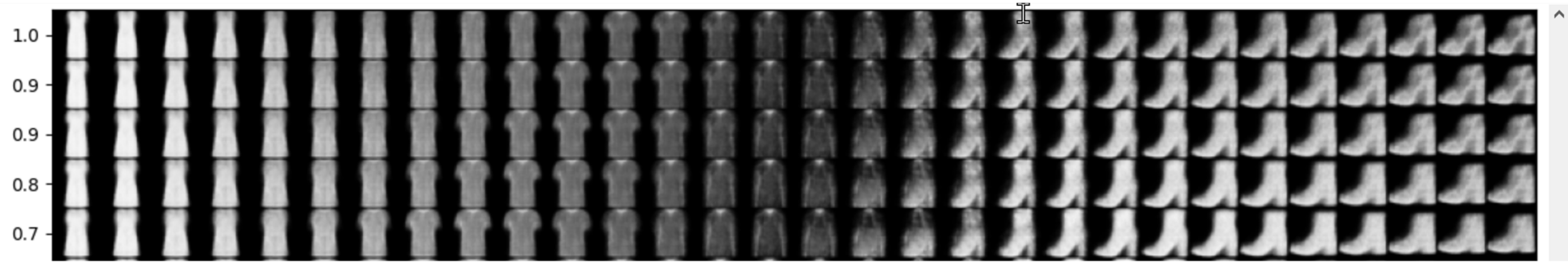
A Variational Autoencoder

DL_033_Autoencoder_VAE



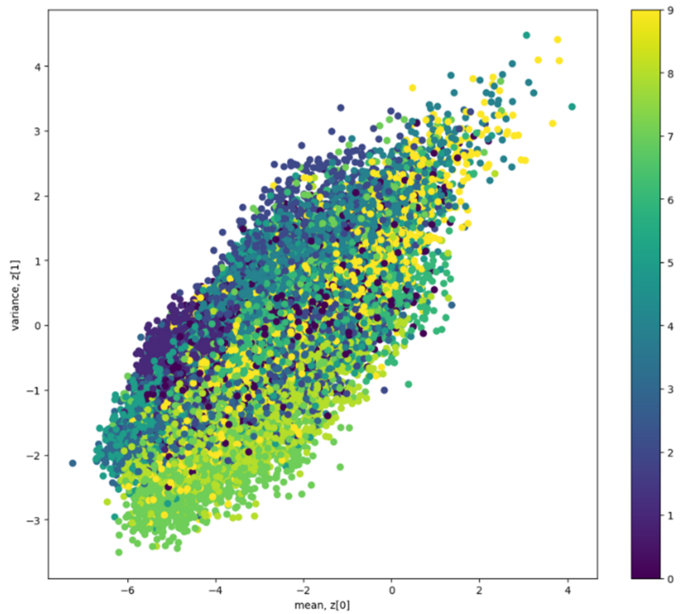

```python
# Build the encoder
def encode(self, data):
    x = self.encoder(data)
    z_mean, z_log_var = self.z_mean(x), self.z_log_var(x)
    return z_mean, z_log_var

# 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

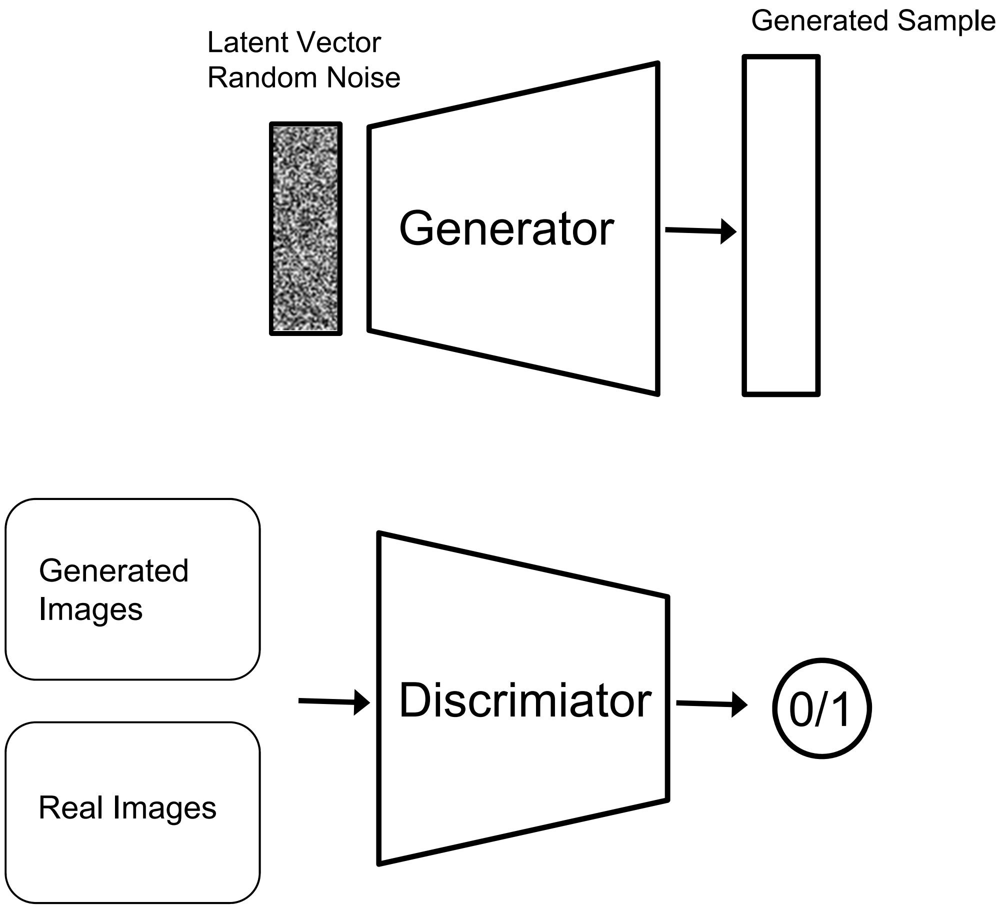
Comments
- Here we have labels very often we autoencode things of the same class that vary just in details



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

Generative Adverserial Networks

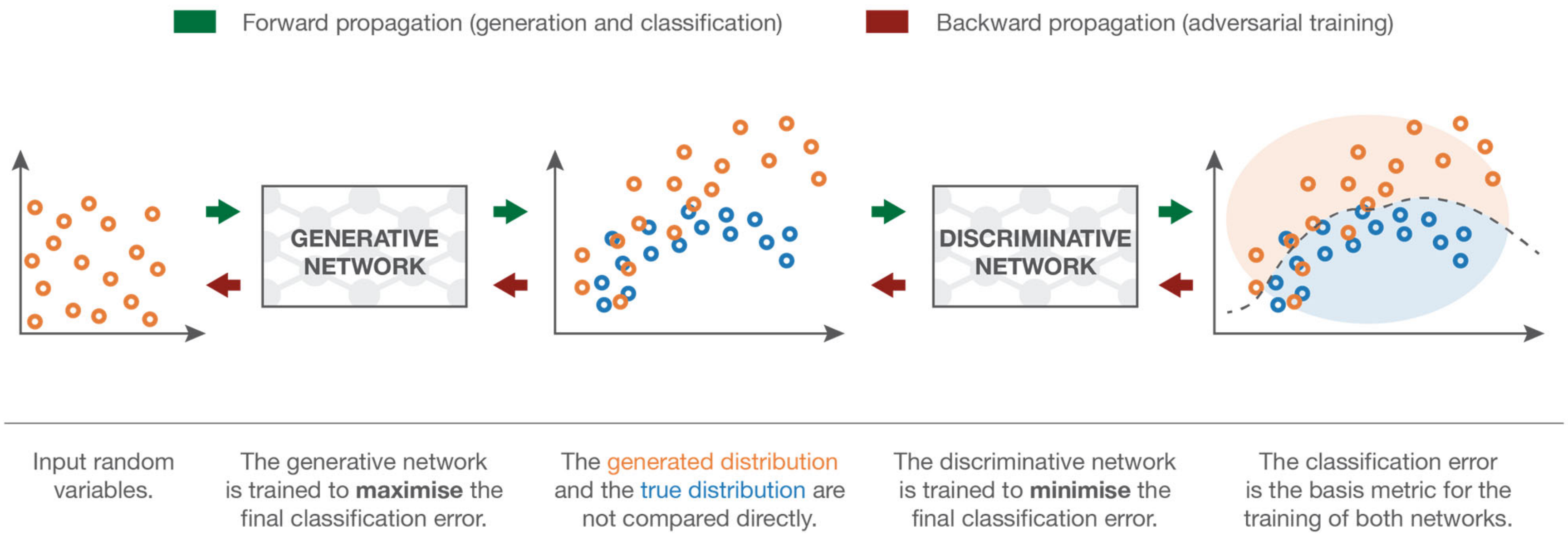
Working Principle of GANs



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.
- The abbreviation “DCGAN” stands for Deep convolutional generative adversarial network

Backpropagation in GANs

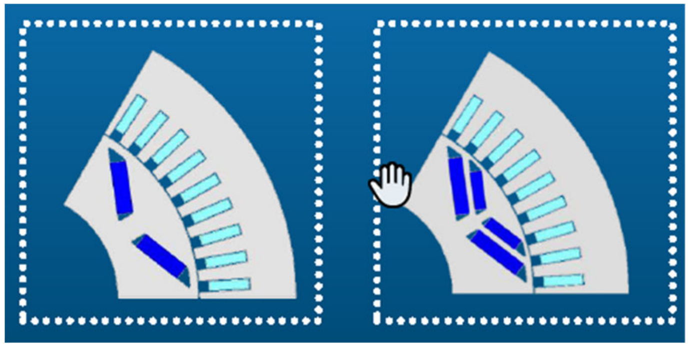


https://www.kaggle.com/code/sayakdasgupta/introduction-to-gans-on-fashion-mnist-dataset

- Both networks are never trained at once instead 2 alternating Phases are used
- Phase I: Discrimiator Training
- Phase 2: Generator Training

Example Applications of GANs

- Generate Realistic Photographs (e.g. Faces)
- Generation of new Motor Designs
- Image-to-Image Translation (e.g. Style GAN 2018)
  - Paintings with a certain Style
  - Face Aging
- Super Resolution (SRGAN 2017)



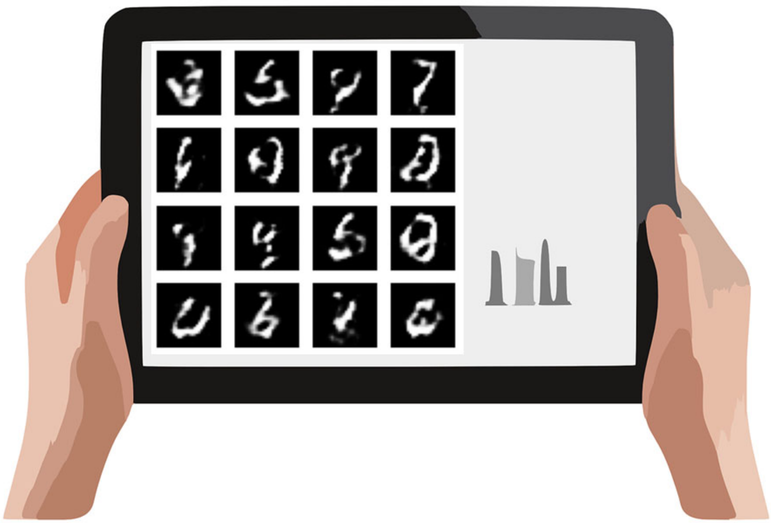
Generative Adversarial Network examples

Fashion MNIST:
https://www.kaggle.com/code/sayakdasgupta/introduction-to-gans-on-fashion-mnist-dataset

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

Task: Play around with The GAN
Given in Tutorial 1 – using
Kaggle Infrastructure

Try to understand how the code
Matches the theory that you saw
on the slides!



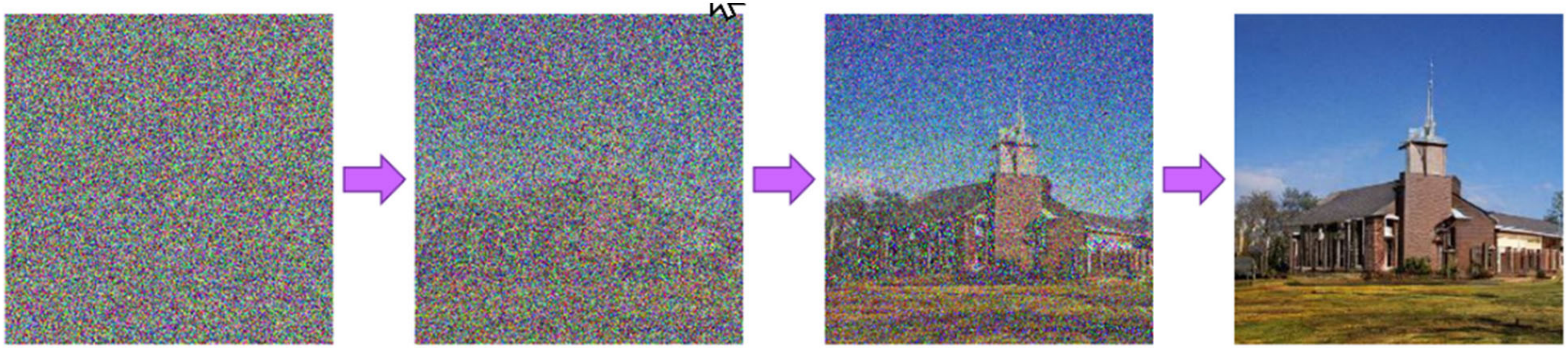
Diffusion models

Diffusion models

- “Denoising Diffusion Probabilistic models” aka DDPM
- That is the case technique behind
  - DALL-E 2 from Open_AI
  - Imagen from Google
- Very effective in generation of photorealistic images
- Can take conditional context into account
- Other Applications
  - Video generation
  - Audio synthesis
  - Text to image
  - …

How diffusion models work

- The network is trained to reduce noise in a picture. This step is repeated until an image is created.
- The generation process can be conditioned with prompts

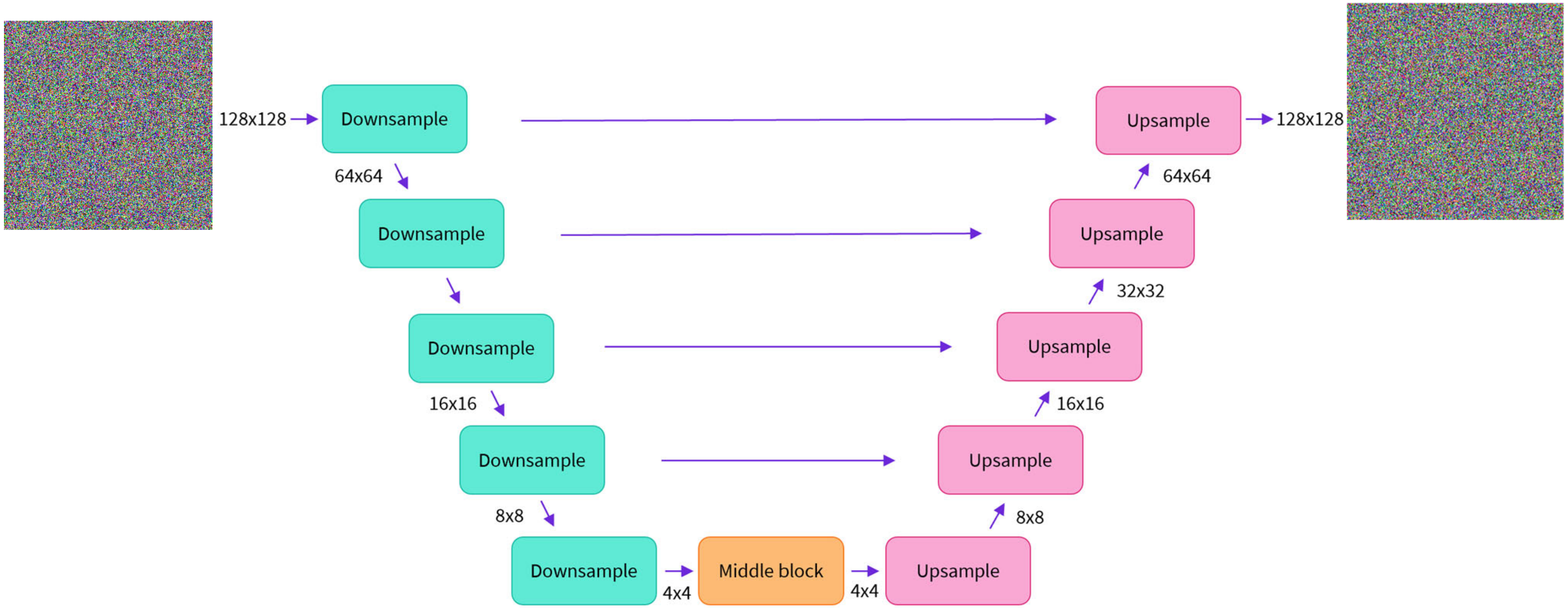


https://colab.research.google.com/github/huggingface/notebooks/blob/main/diffusers/diffusers_intro.ipynb#scrollTo=xkyOEnzuVbsq

NN for Denoising NN for Denoising NN for Denoising

- Two components: A NN for Denoising and a scheduler to control the process.

Unet- CNN-Architecture for Denoising



- Input: Noising Picture + Timestep
- Output: Less noisy image

How diffusion models work

- Typical number of denoising steps is 1000.
- Training data is generated by using images and adding just a little bit of noise each step until we have a completely noisy image
- For generation we just sample a random noise picture and apply the reverse process

How to work with diffusion models

- The popular library “diffusers” (from Huggingface) makes it easy to create and use diffusion models
- A lot of pretrained “ddpm” models are available at huggingface.

google/ddpm-cifar10-32
Updated Nov 8, 2022 · 58.1k · 23

google/ddpm-ema-celebahq-256
Updated Nov 8, 2022 · 10.2k · 2

google/ddpm-cat-256
Updated Nov 8, 2022 · 7.39k · 2

google/ddpm-celebahq-256
Updated Jul 21, 2022 · 5.55k · 18

google/ddpm-ema-bedroom-256
Updated Nov 8, 2022 · 2.93k · 1

google/ddpm-ema-church-256
Updated Nov 8, 2022 · 2.72k · 2

google/ddpm-church-256
Updated Nov 8, 2022 · 1.44k · 5

google/ddpm-bedroom-256
Updated Nov 8, 2022 · 302 · 2

google/ddpm-ema-cat-256
Updated Nov 8, 2022 · 59 · 1

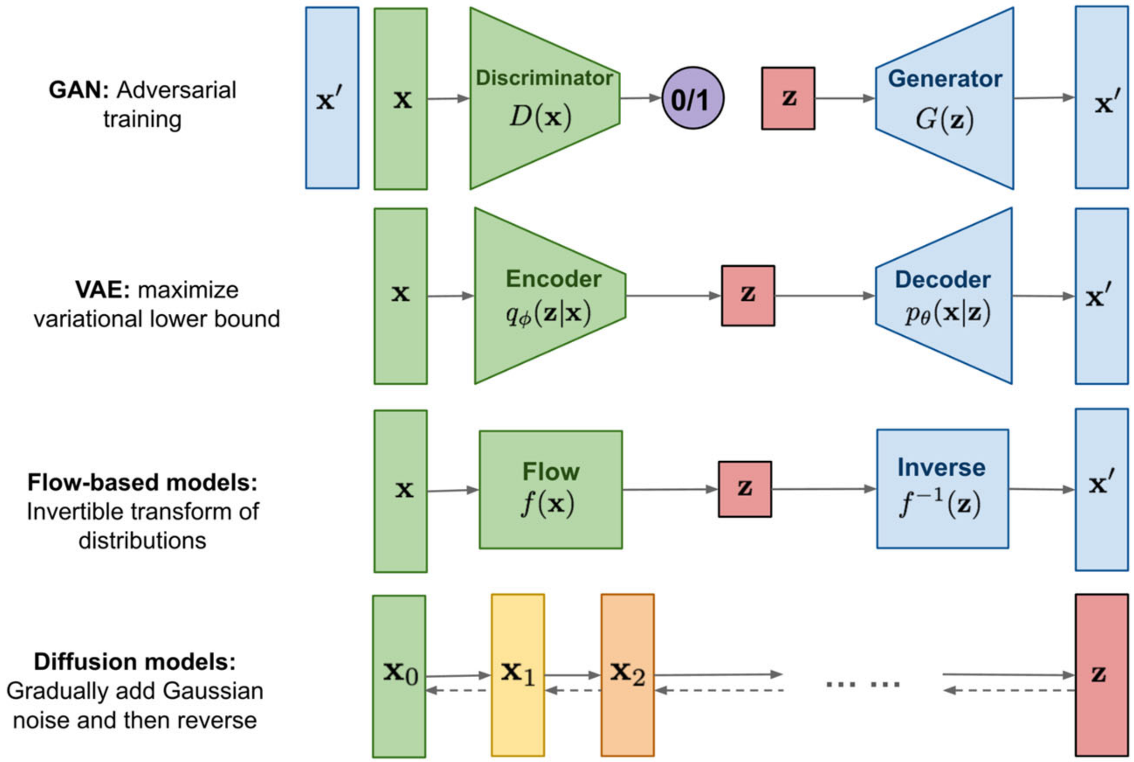
How to work with diffusion models

- Inference Tutorial
  - https://colab.research.google.com/github/huggingface/notebooks/blob/main/diffusers/diffusers_intro.ipynb

```
[1]  from diffusers import DDPMPipeline
     image_pipe = DDPMPipeline.from_pretrained("google/ddpm-celebahq-256")
     image_pipe.to("cuda")
     images = image_pipe().images
```

- Training / FineTune Tutorial
  - https://huggingface.co/docs/diffusers/training/overview
  - https://colab.research.google.com/github/huggingface/notebooks/blob/main/diffusers/training_example.ipynb

Overview on generative models



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

Summary

- Autoencoders consist of an encoder and a decoder. For ech sample they create a smaller representation the latent space. They are rarely applied but were base for other ideas
- VAEs are similar to autoencoder but they allow to generate new samples from random values in the latent space.
- GANs make use of a generator and a discriminator network to learn how to generate/identify artificial images. They are harder to train compared to VAE’s but may produce more realistic pictures.
- DDPMs learn a reversed stepwise denoising process and are thus capable of creating new images from random noise. They are the most recent type of models and outperform GAN’s

Recurrent neural networks
RNN

RNN’s are dealing with Sequences of data

- Sequences are collections of elements, where
  - Elements can be repeated
  - Order matters
  - Sequences can be of variable (up to infinite) length
- Examples
  - Text e.g for translation
  - Audio Signals of variable length for captioning
  - Stock prediction

How did we process text in ML SS2022?

Situation with the NN’s so far….

- Input had fixed length.
- Order of features was not relevant (almost)
- Output had fixed length

- For some Tasks sequence is essential

Example, where bag of words is failing:

“I saw her duck” “Duck! I saw her.”

Overview

- RNN
  - Basic Idea of recurrent networks
- LSTM
  - Advanced version addressing the problems of RNN
- GRU
  - Alternative design to LSTM

Examples variable Inputs and outputs

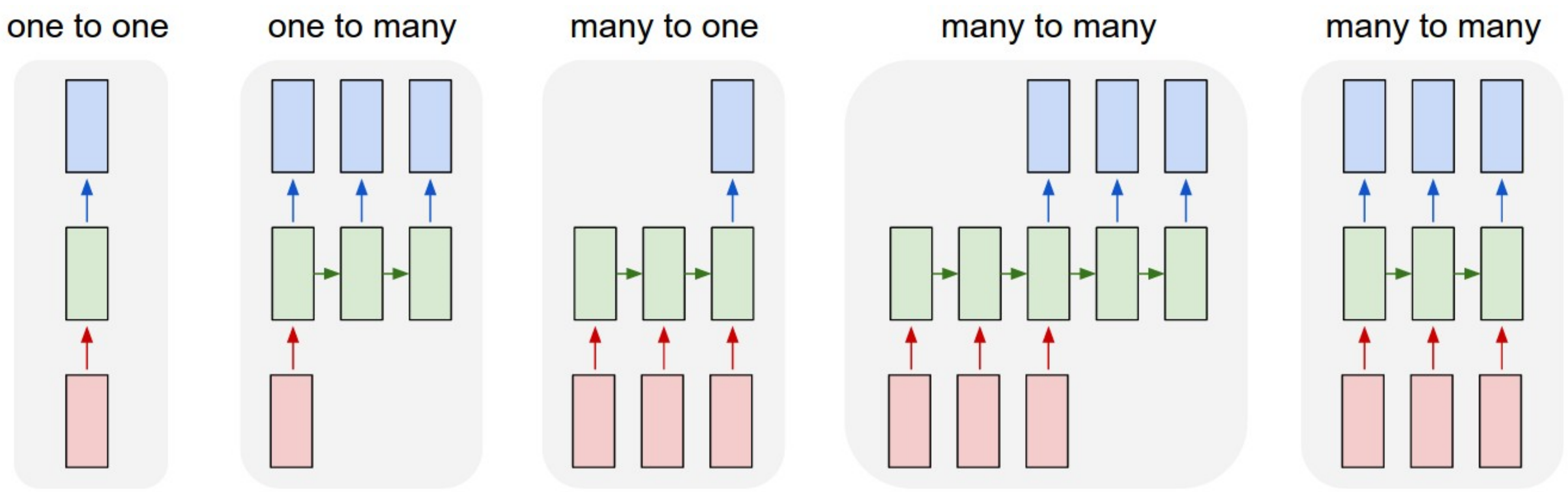


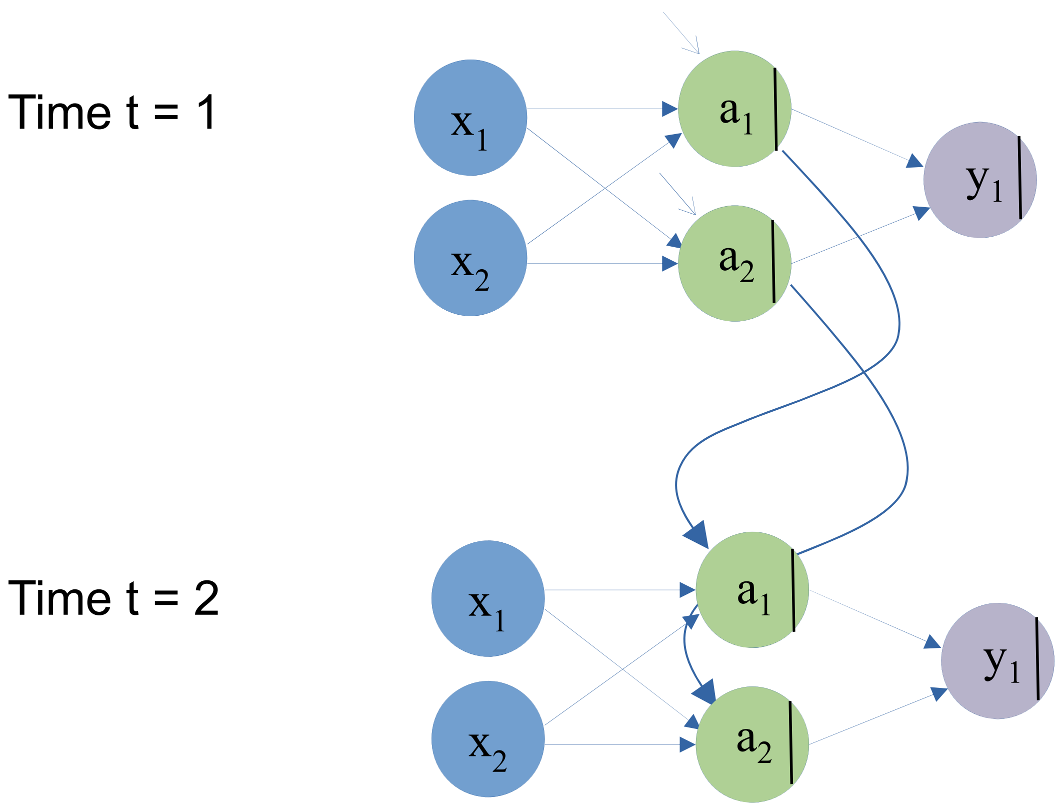
Image Source: http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Text Classification

Image captioning

Translation

Basic recurrent inference

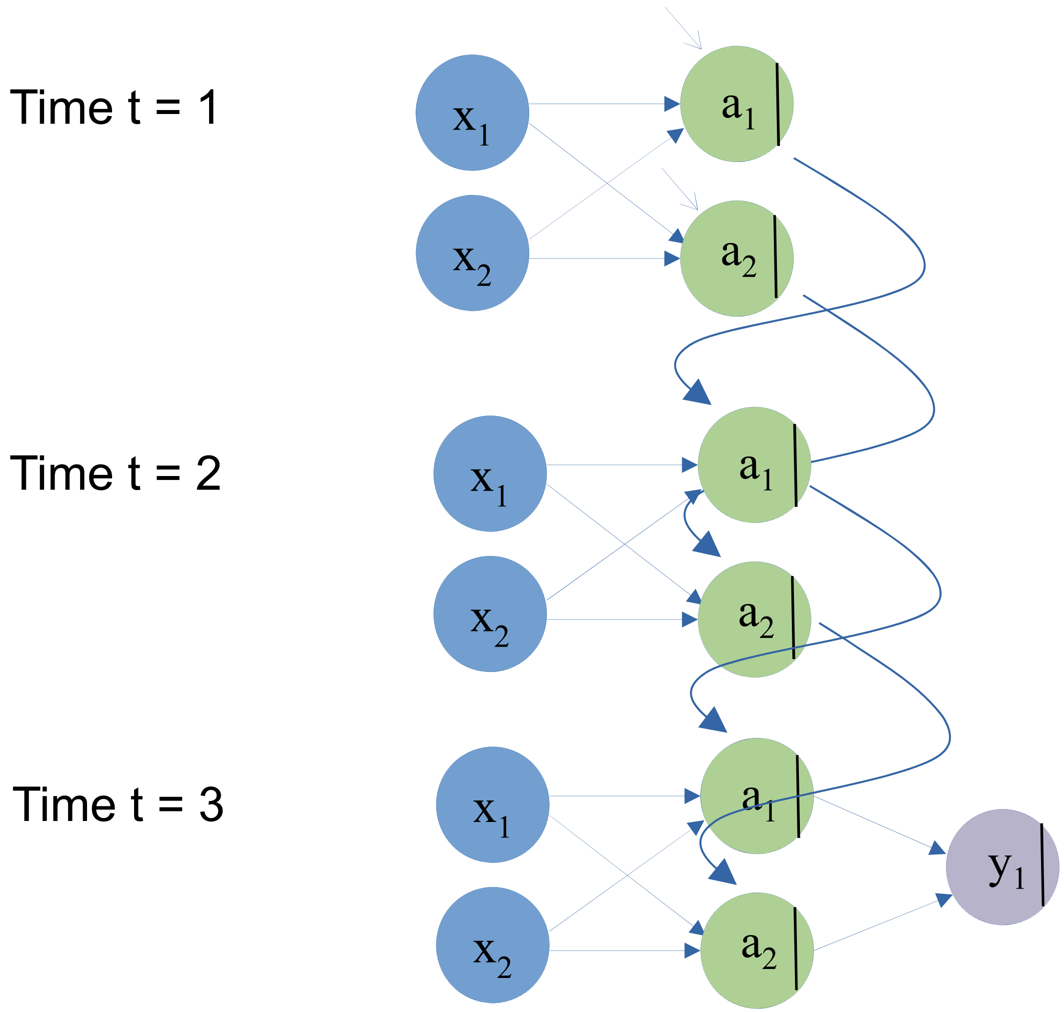


A recurrent layer has Some additional inputs where it gets what he returned as an output the timestep before.

Weights are also given for these recurrent connections!

Initial Value is 0.

Basic recurrent backpropagation



Backpropagation goes Also back in time. This Is similar to very deep networks and leads to the **vanishing gradients problem**

Problems of recurrent neural networks

In theory RNN are powerfully enough to take a long context of samples into account, but in practice its very hard to train these models.

- **Problem 1:** Vanishing gradients, weights cannot be efficiently updated over the long sequence
- **Problem 2:** In a consequence they cannot store a long context efficiently
- **Problem 3:** Slow training process. The training has to be performed sequentially and cannot parallelization is not possible.

The first two problems are addressed by LSTMs.