Exercise Sheet Deep Learning

Part2: Generative Models
Summer 23

This sheet includes a theoretical part and a practical assignment of the second part of the lecture Deep Learning (Generative Models). It should be handed in as pdf in groups of three via ekvv-Moodle until 11.5.23. at 10.a.m (sharp). Please include a link to the code (e.g. Colab)

name1: Elma Nevala

name2: Dovile Umbrasaite

name3: Michel Valentini

PARTI – THEORY: For the following, you might answer only YES/NO (or abstain), or you can add short arguments (at most two lines per question). If you are not sure, it is better to abstain.

1. Training the following generative models relies on the principle . . .

[✓ yes] no maximizing the likelihood for observed data for restricted Boltzmann machines,

[✓ yes] no the reconstruction error of given data for a variational auto-encoder,

[✓ yes] no the direct log likelihood for normalizing flow models,

yes no the direct log-likelihood for diffusion models.

2. The following constraints need to hold for the neural architectures used in . . .

[✓ yes] no diffusion models: invertibility of the implemented function,

yes [✓ no] variational autoencoder: output of the model needs to be low dimensional;

yes **no** GAN: the discriminator has input dimension n, the dimension of the data to be generated;

yes [✓ **no**] normalizing flow: dimensionality stays the same,

3. The following are trainable model parts/parameters of the models ...

yes [✓ **no**] GAN: generator is trainable, discriminator is fixed,

[✓ **yes**] **no** normalizing flow: forward mapping and its approximate inverse mapping,

yes **no** VAE: all trainable parts are encoder and decoder functions, the probabilistic embedding is trained using sampling,

[✓ **yes**] **no** diffusion model: forward and backward mapping are trained,

4. The following holds for GAN variants:

yes [✓ **no**] WGAN optimizes a cost function which is given by the Kullback-Leibler divergence as measures for the difference of distributions,

[✓ **yes**] **no** WGAN CT aims for a Lipschitz continuous discriminator,

yes [✓ **no**] conditional GAN uses a factorized latent space for generating data, where the factors correspond to classes,

yes [✓ **no**] BigGAN provides a text generation module,

5. The following tasks can be addressed by ...

[✓ **yes**] **no** Domain adaptation: Cycle GAN

[✓ **yes**] **no** Diffusion models: to approximate a highly nonlinear probability distribution where sampling is particularly efficient,

[✓ **yes**] **no** Guided diffusion to generate images from labels or signals such as text

yes **no** VAE to have an explicit analytic description of a high dimensional and nonlinear probability distribution

PARTII – PRACTICE: You can use code and models which are publicly available, please clearly reference all sources and tools. Please give a link to your code (e.g. colab). The length of the answer is limited to one page in total for the description of both parts including images. Please provide: short description what you did, how it is done, what is the result. Please be prepared to present the solution in the exercises.

1. Take the Fashion MNIST data set and train a variational autoencoder (VAE) on these data. Provide some insight in how good the data are represented, e.g. by reporting the reconstruction error and displaying typical results of generated images. What happens if you interpolate in between two points in the latent space for the respective encoded data points? Display examples within a class and in between two different classes.

2. Use any other generative model and train it using the Fashion MNIST data. Describe how it is trained and the result.

1 Description Practice

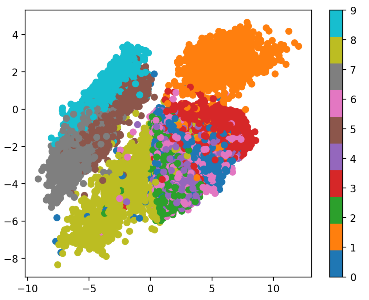
We used the variational autoencoder architechture to generate images based on the fashion MNIST dataset.

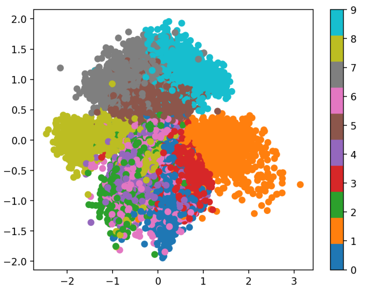
The variational autoencoder achritechture is based on encoding the data into a latent space and decoding it. New images can be generated by interpolating between points in the latent space.

1.1 Fashion MNIST

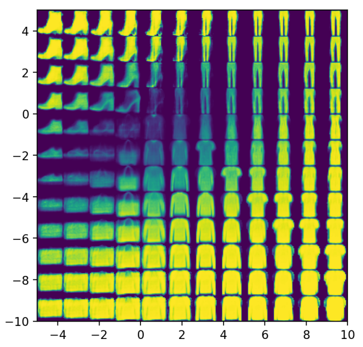
We used pythons pytorch library to create the needed encoders and decoders. We tried two different encoders and provide visualisation for both of them.

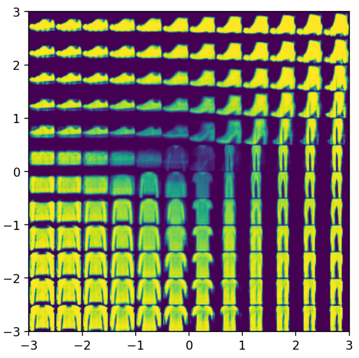
The latent spaces grouped by clothing types:



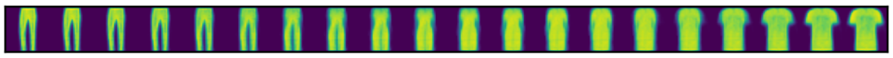


Generated images:





We also interpolated between two classes: trousers and blouses. In the image below you can see the ganarated images between these two classes.



1.2 Link to the Github

https://github.com/ValentiniMichel/deep_Learning_second_assignment/blob/main/second_assignment.ipynb

1.3 Exercise No. 2

To complete the second task of the assignment, a conditional generative adversarial network (cGAN) was implemented. This type of generative model extends the traditional generative adversarial network (GAN) by conditioning the generation process on additional input variables. In a cGAN, both the generator and the discriminator take in additional input variables, called the conditional variables, that are used to guide the generation process.

Overall, the code trains a cGAN to generate images that belong to different classes by learning the underlying distribution of the training data. The generator generates new images that are similar to the training data and the discriminator distinguishes between real and fake images. By training these two networks together, the cGAN learns to generate realistic images that are similar to the training data.

The received results are presented in the generated image below: