Practical Work 13 – 19/05/2022
Generative Models : VAE and GAN

Objectives

The objective of this PW is to experiment with generative AEs, VAEs and GANs. You can do the exercise using your favorite framework, either with Tensorflow or Pytorch.

Submission

— **Deadline** : Wednesday, 1st of June, 12am (noon)

— **Format** : Zip with your jupyter notebooks.

Exercise 1 Autoencoders and Variational Autoencoders

Plain Autoencoders

Implement different versions of a "plain" auto-encoder with 128 latent dimensions, at minimum

a) one with only fully connected layers,

b) one with only CNNs.

Train them with Fashion-MNIST. Track the reconstruction error (train and test per epoch).

What model leads to the best (smallest) reconstruction error on the test set ? Verify that also by looking at individual samples from the (test) set.

Explore the latent space of the trained models by constructing images from latent variables as follows :

a) Pick a random image and, by encoding it, map it to the latent space (variable z). Then look at the resulting reconstructions from scaled z-variables, i.e. $\tilde{z}(t) = t\,z, t > 0$.

b) Linearly interpolate between latent variables associated with encodings of two arbitrarily selected images and reconstructing them.

c) Reconstruct from randomly sample latent variables, sampled from a d-dim standard normal distribution.

Hint : For the CNNs you may first start with the exercise with GANs below.

Variational Autoencoder

Transform your CNN-only AE from the previous subsection into a VAE, again using 128 latent dimensions. During training track both, the reconstruction error and the KL-loss. Could you improve the reconstruction error by when comparing with the plain autoencoder ?

Explore the latent space of the trained VAE - following the same steps as above. Describe and judge/interpret of what you observe. How does the random generation by sampling from the latent space work now ?

Exercise 2 Generative Adversarial Networks : DCGAN

Start from the DCGAN tutorial (for pytorch or tensorflow).

a) Go in detail through the models for the Generator and Discriminator, compute the dimensions of the activation maps for each of the layers - given the filter size, padding and in/out channels. Checks how the tensor shapes are transformed into the shape in latent space.

b) Then, adjust the code to make it work with Fashion-MNIST and train the GAN.

c) Keep track of the generator and the discriminator loss. Can you observe stable behaviour ?

d) Periodically, create a mosaic plot (with 8x8 images) generated the same from randomly sampled latent variables. This allows you to visually track progress of how the training makes progress in learning the distribution and how to generate new FashionMNIST-like images.

e) Play with the hyper parameters (e.g. learning rate) to see whether you can see yet the same stable behaviour.