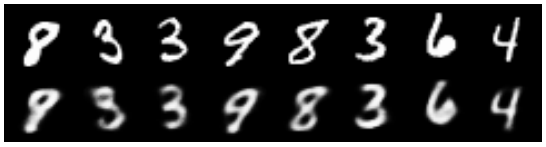
This file include problem 4.2, 4.3, 5.3, 5,5.

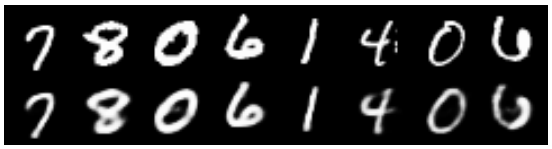
Problem 4 default hyperparameters

- Batch-size: 128
- Epochs: 10
- No-CUDA: False
- Seed: 1
- Log-intervals: 10

Problem 4.2



Reconstruction sample for the **first** epoch on the MNIST dataset using default hyperparameters



Reconstruction sample for the **last** epoch on the MNIST dataset using default hyperparameters

Problem 4.3

Reconstruction sample for the **first** epoch on the Fashion MNIST dataset using default hyperparameters

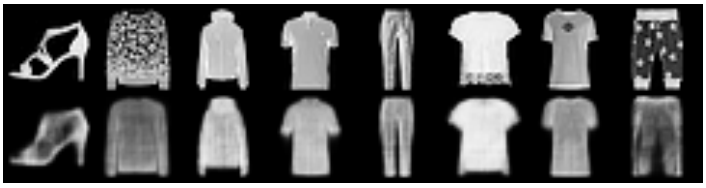
Reconstruction sample for the **last** epoch on the Fashion MNIST dataset using default hyperparameters

We first fixed epoch = 10 and experimented using batch size [64, 128 (default), 256]

batch-size = 64, epochs = 10

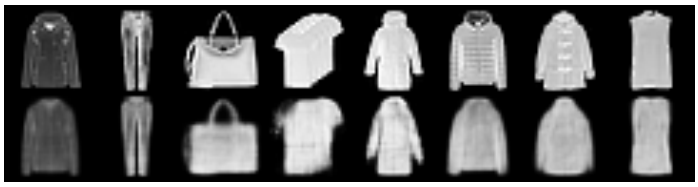
using batch-size = 256, epochs = 10

We can see that when batch size = 64, it is slight better on the shoe reconstruction. But all the batch sizes we used gave more or less similar results. We then tried batch_size = [64, 128] and different epochs



batch-size = 64, epochs = 20

batch-size = 64, epochs = 50

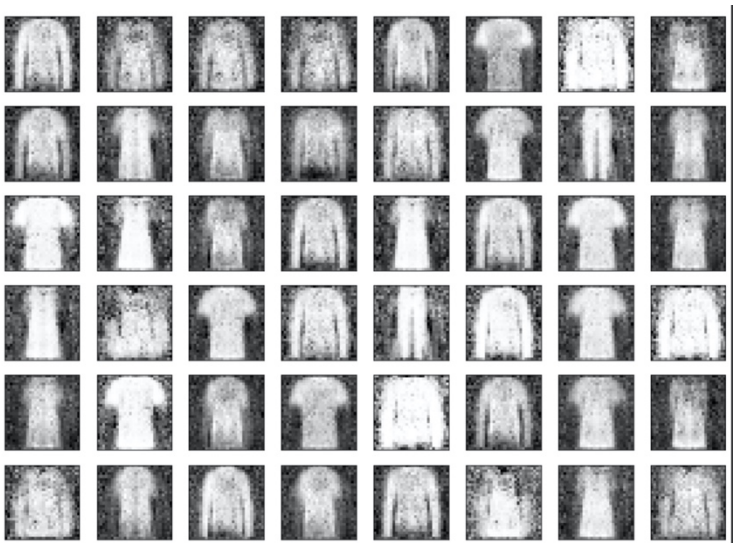


batch-size = 128, epochs = 300

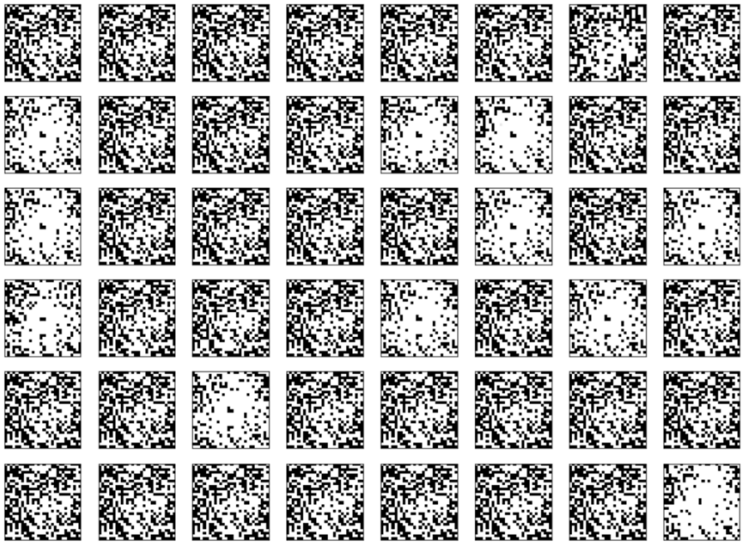
It seems the batch size of 64 with 50 epochs performs the best. This can be seen in the logo of the shirt being reconstructed, the bags and shoes don't seem to be great in any of the examples, but with the batch size of 64 and 50 epochs we are starting to see the presence of smaller details.

Problem 5.3

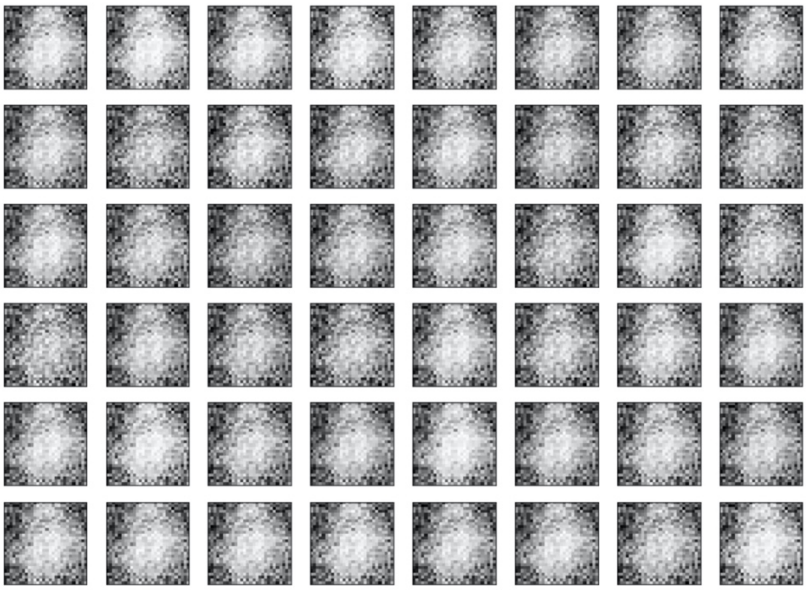
GAN equal LR - We see pixelated outlines of some of the clothing items. Most are shirts, dresses and in some cases pants. This seems to be the most optimal training setup.



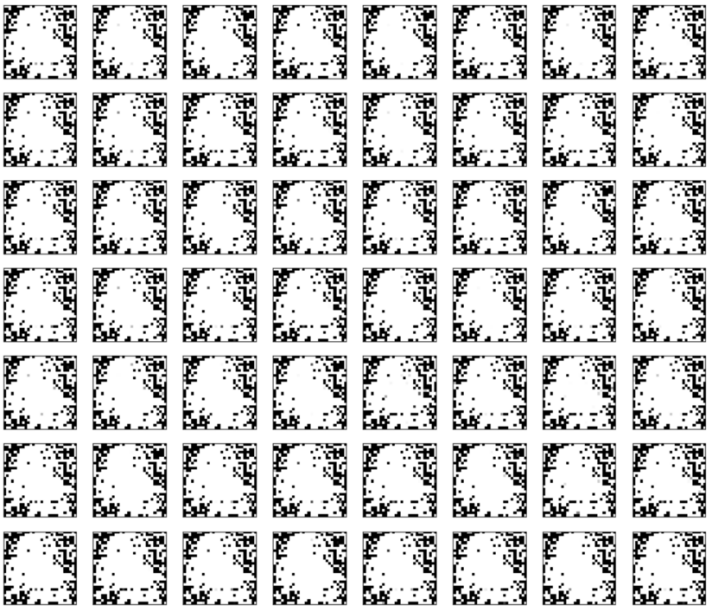
GAN Generator LR (0.002) > Discriminator LR (0.0002) - The generator loss was very high and it seems like it was never able to learn to “trick” the discriminator.



GAN Generator LR (0.0002) < Discriminator LR (0.002) - The generator loss was always 0 and the discriminator loss was very high. It seems like the discriminator was unable to determine what was real and fake, so the generator was able to create blobs and “fool” the discriminator.



GAN Generator training 3x for each Discriminator train - There is an outline of a blob forming, but the generator loss was too high whereas the discriminator loss was near 0, similar to the generator learning rate being greater than the discriminator learning rate



Problem 5.5

CGAN - We see very fuzzy images with matching labels. The images seem to be less clear than the vanilla GAN but do have the added benefit of being able to determine the label. However, it does seem like the CGAN was able to get a better distribution of products. In the vanilla GAN, there were only shirts, dresses, and trousers, most likely because they all have a somewhat similar shape. In the CGAN, the generator was forced to learn more categories because of the labels, which is most likely why there is such a decrease in image quality.

Problem 5.6

What differences did you notice in practice? (I have put the following answer in the notebook as well)

The images reconstructed from the VAE seem to be much more clear. However, with the GANs we can see a design starting to form in the clothing, something that the VAE doesn't do particularly well. In the case of the GAN, it seems even the background is not learned to be a solid color, which is something the VAE does very well. In an applied setting, each has their own merits. If one is trying to get new clothing ideas, a GAN would be better, particularly a cGAN, as you can specify the category to create.