CS764: Assignment 8

Report

Siddharth Saha 170100025 **Parth Shettiwar** 170070021 **Parikshit Bansal** 170050040

Part A

Explain the nomenclature DC

DC here means Deep Convolutional. The original GANs were proposed with FeedForward Linear Layers, but for image datasets Deep convolutional Generator and Discriminator performed better hence they became the standard for images.

What is the probability distribution of the output images generated by the GAN? Are all digits generated with equal probability? Speculate.

No not all digits are generated with equal probability. The claim is also supported by the samples in 1 which doesn't seem to have 5. We also tried drawing multiple such batches of 40 images and concluded that there is a mismatch between input (uniform) and generator distribution. There might be 2 reasons for the same

1. Digits which are harder to generate like 5 are easily identified as fake by the discriminator (given that it learns the features of 5) as compared to simple digits like 1 which is just a straight line. This might be causing the Generator network to converge on a locally optimum solution where it doesn't produce digits like 5 at all. There's a tradeoff to be considered here though. If the generator just produces images of a single digit like say 1, then discriminator can with high probability identify fake vs real, as it is just a image classification problem with 1's being MAJORLY fake and all others digits being always real

2. The other issue is that NNs don't look at digits as a whole but instead look at the features of these digits. For example 1 and 7 both have a straigth like as a feature, 3 and 8 have similar structure, 6,8,0 are also quite similar in features like circles. Since these features like circles and lines are more common in the dataset (as they occur across digits, compared to less common unique structures like 4 and 5), they are better learnt by the generator and are outputed hence more frequently

Part B

What is tricky about the original GAN? Write in your own words making sure you understand the buzz words. Try to draw a picture if that helps the explanation.

One of the major issues of the Original GAN was with Vanishing Gradients i.e. low gradient signal for generator to facilitate training. This is due to the fact that the output of discriminator is always a number between 0 and 1 in original GANs and it output 0/1 with high probability. This is due to the high confidence of discriminator in its judgement. But these signals of 0/1 are not enough for

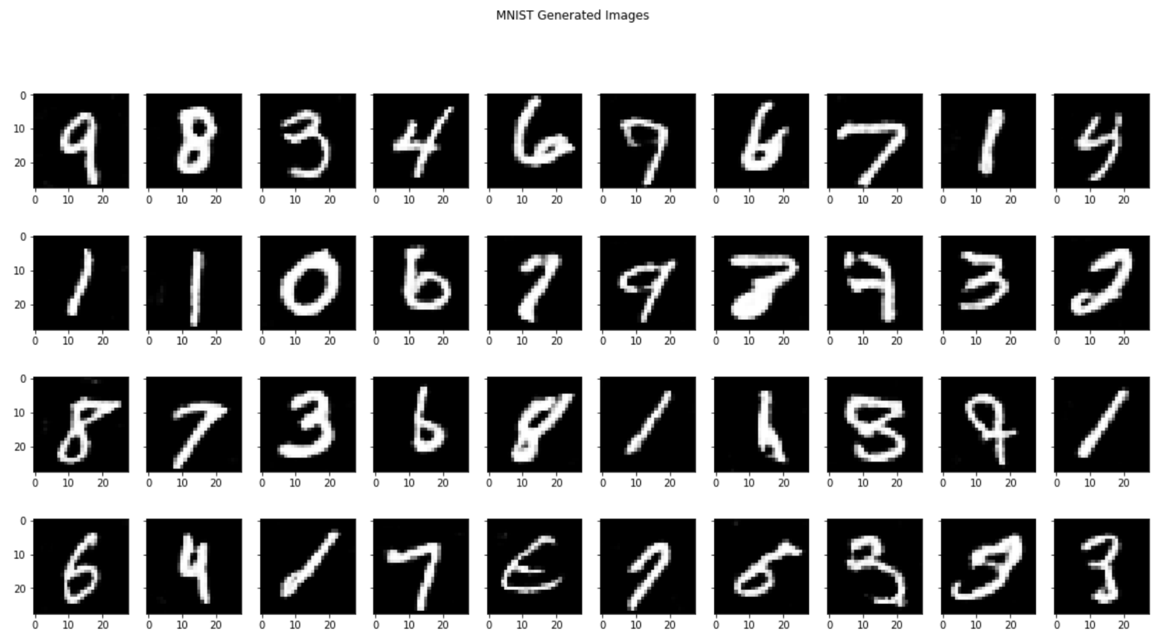

Figure 1: Caption

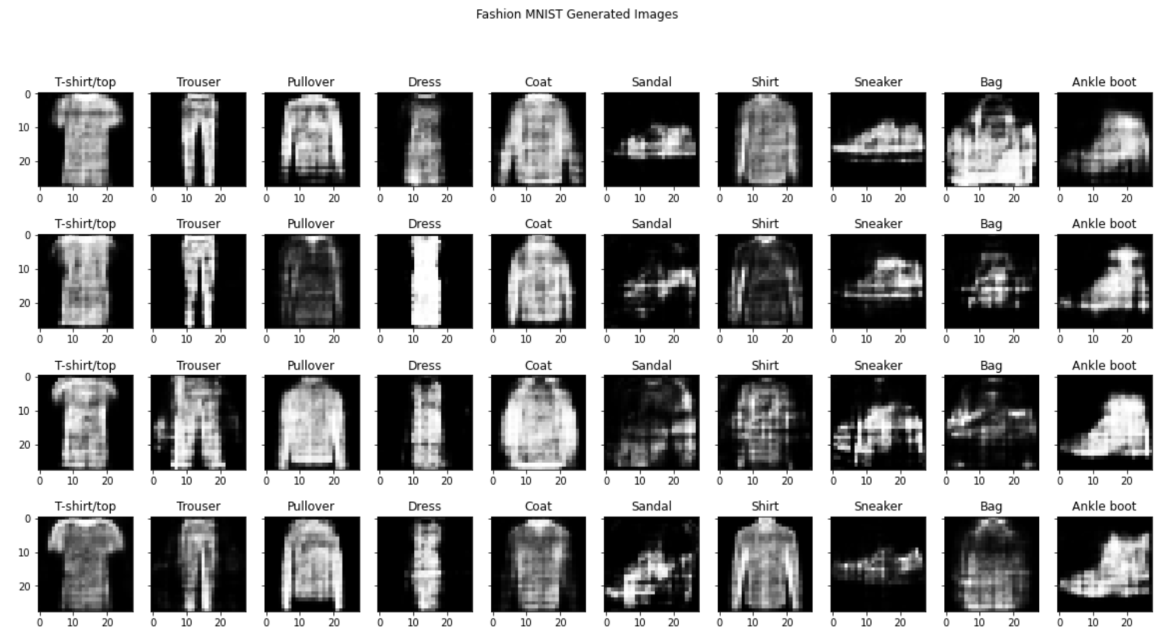

Figure 2: Caption

the generator to train well. That is they lie in the saturation zone of the sigmoid. This can also be compared to how ReLU replaced sigmoid as the activation functions due to better gradient flow.

Moving away from outputing probabilities can also be seen as discriminator now ”Critiqueing” or scoring Generators forgery instead of just classifying it as fake or real. So its akin to giving detailed score report to the student, instead of just a pass/fail grade, the later not giving student the idea of his shortcomings.

Assume your assignment is successful. How would you use this in a business, real world setting?

GANs are one of the state of the art unsupervised image generative models surpasing their predecessor VAEs. Unsupervised learning methods are a hot topic of discussion these days due to the abundance of raw images online but scarcity of labelled data for the saem. These methods hence can learn usefull latents from images, which in case of GANs are these 100 dimensional vectors. These vectors have additive properties like if we have 2 latents one which produces an image wearing a hat and another wearing sunglasses, the addition of these vectors will produce images wearing both sunglasses and hat. These things are quite fascinating to be done in image editing software and filters

However GANs suffer some major challenges too. Unlike VAEs and Normalising Flows, GANs can output a hidden representation i.e. latents given input image. This limits the use case of GANs in setups likes representation learning, or image compression where these latents of images are important for the downstream task

Describe your process of conditioning.

Unlike DCGAN where the Generator and Discriminator are both Deep Convolution based NNs, here we move to simple FeedForward Networks instead. For low res images likes FashionMNIST, Linear Layers work quite well also hence model performance was not an issue. The main reason for this move was better conditioning. On treatingg the image as a flattened vector, we can just append a 10 size vector to our latent sample of size 100 to get a 110 size vector which has the information of class encoded into it in last 10 dimensions. Also to our 28*28 size vector input to Discriminator we can similarly append size 10 vector to get 28*28+10 dimensional vector which can be appropriately judged by the discriminator. This is much more efficient than the 2d version counter part where we would require to have an input of 28*28*11 to the discriminator.

Other Details

Both the parts are trained for 100 epochs, using Adam Optimiser with learning rate of 1e-4 which seemed to work the best. Also detaches have been made at appropriate places and scores instead of probabilities have been outputed for WGANs. Noise used for sampling is Standard Normal Distribution. The architectures are heavily inspired from public github repos in the references [1, 2, 3]. Training is GPU compatible.

References

[1] https://github.com/AKASHKADEL/dcgan-mnist/blob/master/networks.py

[2] https://github.com/arturml/mnist-cgan

[3] https://github.com/Zeleni9/pytorch-wgan

[4] https://pytorch.org/docs/stable/nn.html