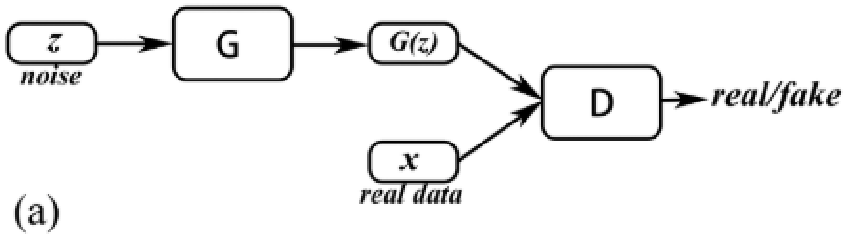
EP24B038 S Sharan Keshav

AI Club, CFI IITM

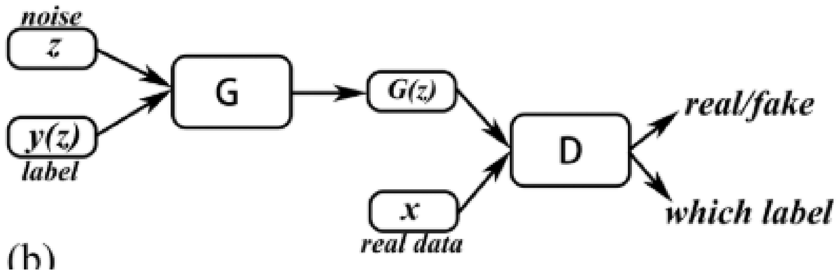
A Brief Summary of Conditional Generative Adversarial Networks

Introduction :

The **Conditional Generative Adversarial Network** architecture improves on the GAN architecture, which allows for image generation without the need to approximate the training distribution or include Markov chains-avoiding a lot of math which in the real world is not completely calculable. cGANs now give us the ability to tailor our generated output based on input class labels.



GAN Architecture



cGAN Architecture

Architectural nuances :

The Generator (G) now takes 2 inputs, a random noise vector, usually sampled by a standard normal distribution of the space z being modelled by G to match x, the original distribution, and a class label, y. These inputs are taken in by G and combined into a joint hidden representation, thus allowing flexibility in labelling.

$$\min_G \max_D V(D, G) = \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x}|\boldsymbol{y})] + \mathbb{E}_{\boldsymbol{z} \sim p_z(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z}|\boldsymbol{y})))]$$

Loss function

The loss function for both the generator and discriminator is modified so as to penalise both D and G for accepting and generating outputs of classes other than the one being dealt with respectively.

When a GAN generates an image, one has no control over the generated image, only that it mimics one of the classes of objects from the training data. The conditioning on class labels during training in cGANs now allows us to generate images of the specific class we desire by passing an input to while generating.

Multiple problems with the architecture of the generator have been fixed by the addition of class labels-

a. Output class is now controllable and not randomly generated.
b. Mode collapse is now reduced because the generator is penalised for learning each class' representation better.
c. The output of discriminator D can be made a probability distribution over all classes, allowing D to predict the class label as well.

The adversarial game remains mostly unchanged, with D and G playing a minimax game until a point where the generator images are good enough that the discriminator is unable to tell them apart from real ones (given the class). At this point, we have for ourselves, a good Generator G, now with the ability to generate, given a class, something that was previously not possible.

Conclusion :

cGAN architecture simultaneously trains two feedforward neural networks, via a minimax game, with the generator trying to capture the underlying data distribution given a label, and the discriminator judging whether the sample given its class came from the training dataset or the generator. This process continually improves the generator, leading to more realistic synthetic images.