GAN

How does GAN Work?

As we've discussed that GANs consists of two ANN or CNN models:

1. Generator Model: Used to generate new images which look like real images.
2. Discriminator Model: Used to classify images as real or fake.

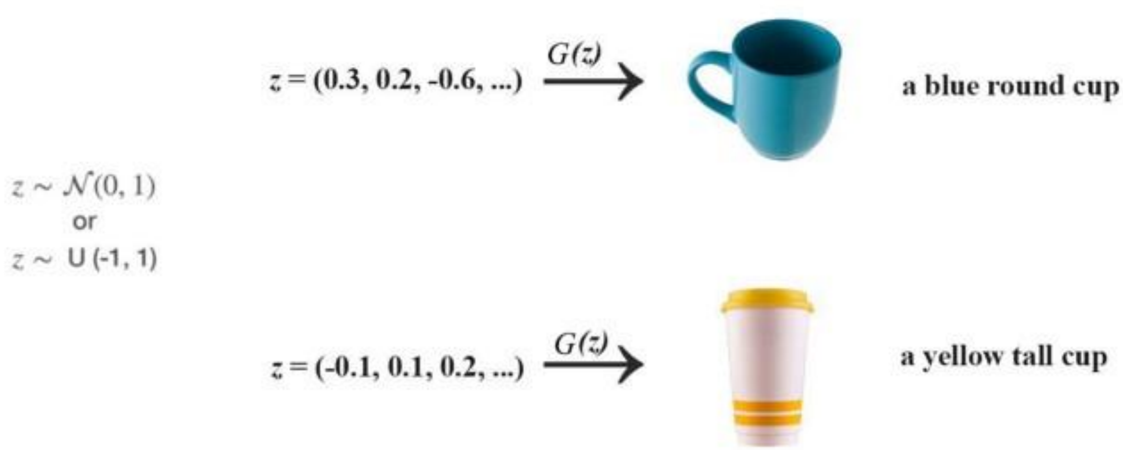
Let us understand each separately.

Note: For simplicity, we'll consider the Image Generation application to understand the GANs. Similar concepts can be applied to other applications.

The Generator Model

The Generator Model generates new images by taking a fixed size random noise as an input. Generated images are then fed to the Discriminator Model.

The main goal of the Generator is to fool the Discriminator by generating images that look like real images and thus makes it harder for the Discriminator to classify images as real or fake.

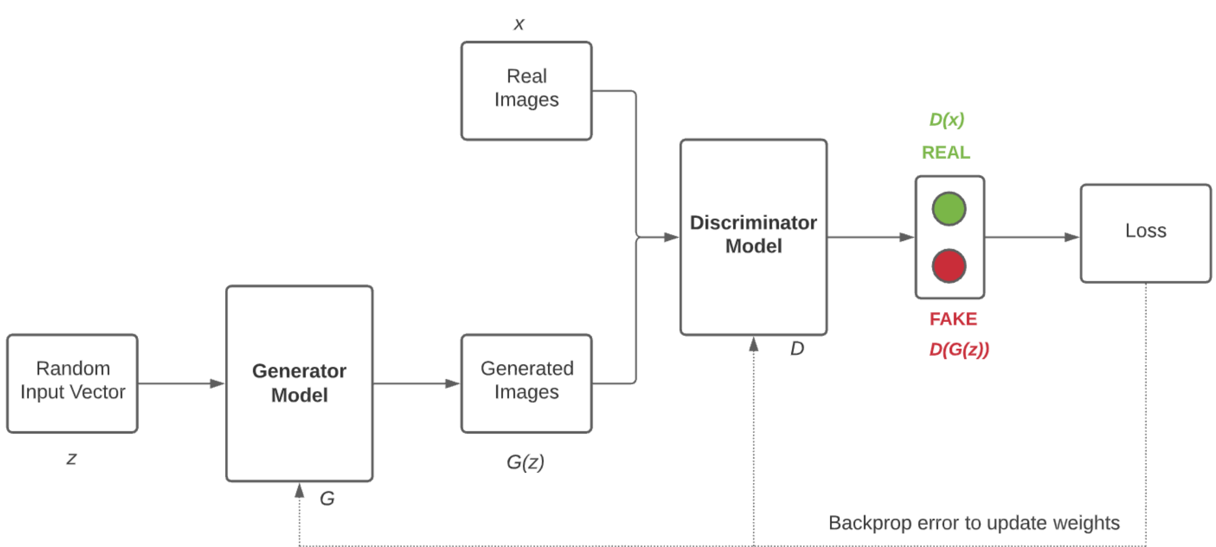


The Discriminator Model takes an image as an input (generated and real) and classifies it as real or fake.

Generated images come from the Generator and the real images come from the training data.

The discriminator model is the simple binary classification model.

Now, let us combine both the architectures and understand them in detail.



The Generator Model *G* takes a random input vector *z* as an input and generates the images *G(z)*. These generated images along with the real images *x* from training data are then fed to the Discriminator Model *D*. The Discriminator Model then classifies the images as real or fake. Then, we have to measure the loss and this loss has to be back propagated to update the weights of the Generator and the Discriminator.

When we are training the Discriminator, we have to freeze the Generator and back propagate errors to only update the Discriminator.

When we are training the Generator, we have to freeze the Discriminator and back propagate errors to only update the Generator.

Thus the Generator Model and the Discriminator Model getting better and better at each epoch.

We have to stop training when it attains the Nash Equilibrium or *D(x) = 0.5 for all x*. In simple words, when the generated images look almost like real images.

Let us introduce some notations to understand the loss function of the GANs.

G	Generator Model
D	Discriminator Model
z	Random Noise (Fixed size input vector)
x	Real Image
G(z)	Image generated by Generator (Fake Image)
$p_{data}(x)$	Probability Distribution of Real Images

$p_z(z)$	Probability Distribution of Fake Images
$D(G(z))$	Discriminator's output when the generated image is an input
$D(x)$	Discriminator's output when the real image is an input

The fight between the Generator Model and the Discriminator Model can be expressed mathematically as:

$$\min_{G} \max_{D} V(D, G)$$

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

Note: The term $E_{x \sim pdata(x)}$ [log D(x)] can be read as E of log(D(x)) when x is sampled from $p_{data}(x)$ and similar for the second term.

As we can see in the equation, the Generator wants to minimize the *V(D, G)* whereas the Discriminator wants to maximize the *V(D, G)*. Let us understand both terms:

1. $E_{x \sim pdata(x)}$ [log D(x)]: Average log probability of D when real image is input.
2. $E_{z \sim pz(z)}$ [log(1 – D(G(z)))]: Average log probability of D when the generated image is input.

Let us understand the equation by thinking from the Generator's and the Discriminator's perspectives separately.

Discriminator’s perspective

The Discriminator wants to maximize the loss function *V(D, G)* by correctly classifying real and fake images.

The first term suggests that the Discriminator wants to make D(x) as close to 1 as possible, i.e. correctly classifying real images as real.

The second term suggests that the Discriminator wants to make D(G(x)) as close to 0 as possible, i.e. correctly classifying fake images as fake and thus maximize the term eventually (1 – smaller number will result in a larger number). *Note: Probability lies in the range of 0-1.*

Thus, The Discriminator tries to maximize both the terms.

Generator’s perspective

The Generator wants to minimize the loss function *V(D, G)* by generating images that look like real images and tries to fool the Discriminator.

The second term suggests that the Generator wants to make D(G(z)) as close to 1 (instead of 0) as possible and thus minimize the term eventually (1 – larger number will result in a smaller number). So that the Discriminator fails and misclassifies the images.

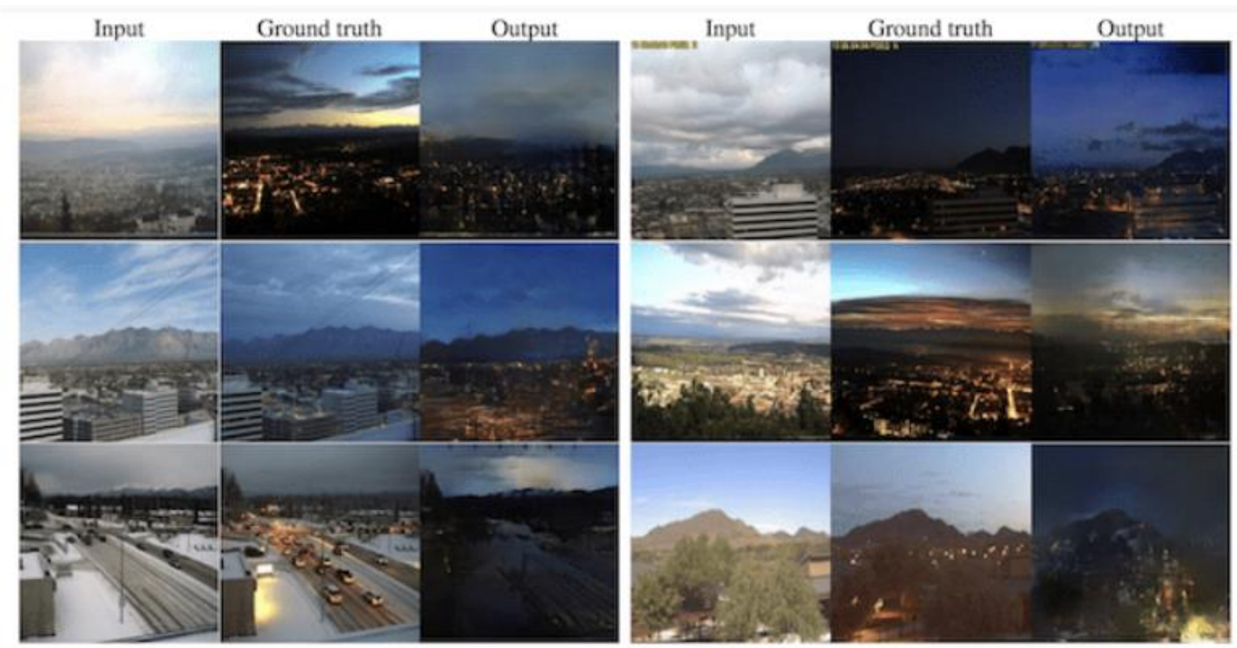
Thus, The Generator tries to minimize the second term.

Amazing Applications of GAN

Let us discuss some amazing applications of GANs other than image generation.

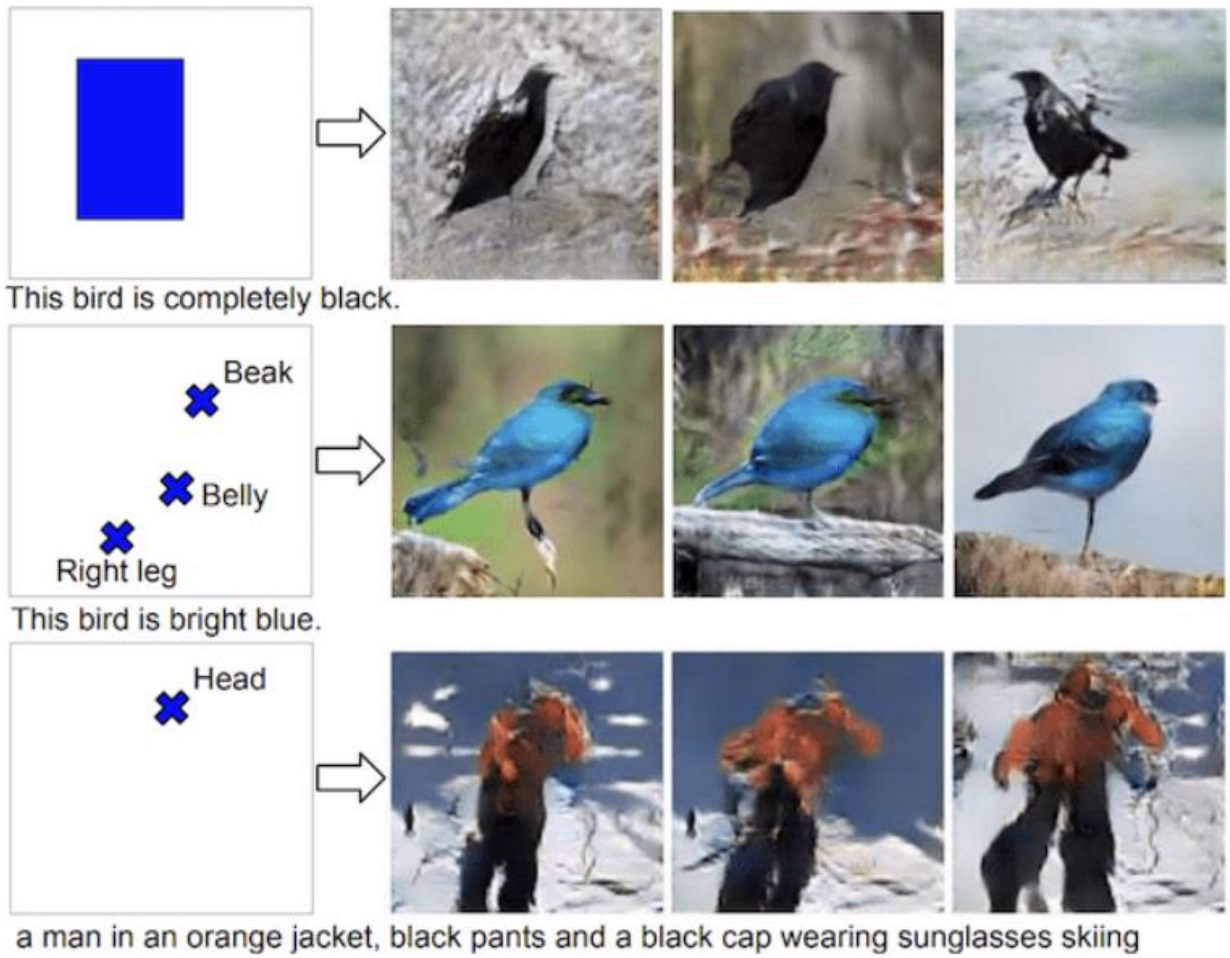
Image to Image Translation

Phillip Isola, et al. in this paper demonstrates GANs as many images to image translation tasks.



Text to Image Translation

Scott Reed, et al. in this paper, demonstrates a way to generate images from text.



https://www.analyticsvidhya.com/blog/2021/06/a-detailed-explanation-of-gan-with-implementation-using-tensorflow-and-keras/

https://jonathan-hui.medium.com/gan-whats-generative-adversarial-networks-and-its-application-f39ed278ef09

https://medium.com/deep-math-machine-learning-ai/ch-14-1-types-of-gans-with-math-5b0dbc1a491d

Lab

https://machinelearningmastery.com/how-to-develop-a-generative-adversarial-network-for-an-mnist-handwritten-digits-from-scratch-in-keras/

https://machinelearningmastery.com/impressive-applications-of-generative-adversarial-networks/