**Module 19**

Explain the detailed steps to train a GAN model. Also explain on the architecture of SRGAN

### Ans: Steps to train a GAN

**Step 1: Define the problem.**Do you want to generate fake images or fake text. Here you should completely define the problem and collect data for it.

**Step 2: Define architecture of GAN.**Define how your GAN should look like. Should both your generator and discriminator be multi layer perceptrons, or convolutional neural networks? This step will depend on what problem you are trying to solve.

**Step 3: Train Discriminator on real data for n epochs.**Get the data you want to generate fake on and train the discriminator to correctly predict them as real. Here value n can be any natural number between 1 and infinity.

**Step 4: Generate fake inputs for generator and train discriminator on fake data.**Get generated data and let the discriminator correctly predict them as fake.

**Step 5: Train generator with the output of discriminator.**Now when the discriminator is trained, you can get its predictions and use it as an objective for training the generator. Train the generator to fool the discriminator.

**Step 6: Repeat step 3 to step 5 for a few epochs.**

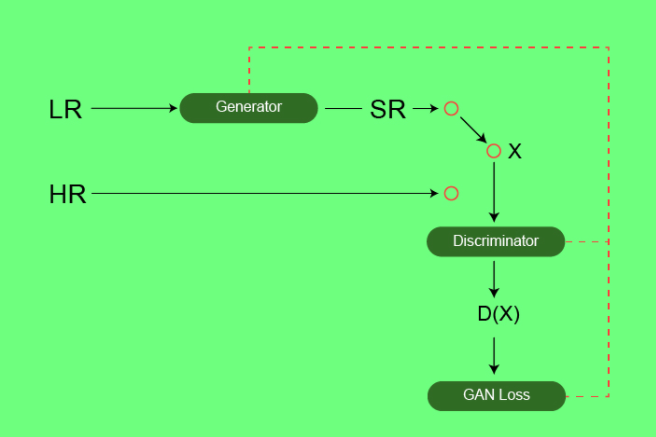
**Step 7: Check if the fake data manually if it seems legit. If it seems appropriate, stop training, else go to step 3.**This is a bit of a manual task, as hand evaluating the data is the best way to check the fakeness. When this step is over, you can evaluate whether the GAN is performing well enough.

Now just take a breath and look at what kind of implications this technique could have. If hypothetically you had a fully functional generator, you can duplicate almost anything. To give you examples, you can generate fake news; create books and novels with unimaginable stories; on call support and much more. You can have artificial intelligence as close to reality; a true artificial  intelligence! That’s the dream!!

# Super Resolution GAN (SRGAN)

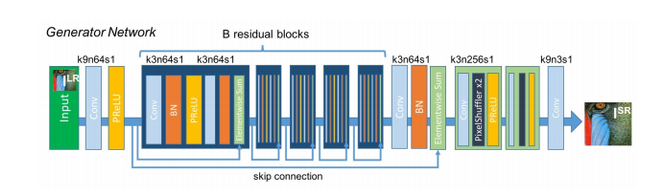
SRGAN was proposed by researchers at Twitter. The motive of this architecture is to recover finer textures from the image when we upscale it so that it’s quality cannot be compromised. There are other methods such as Bilinear Interpolation that can be used to perform this task but they suffer from image information loss and smoothing. In this paper, the authors proposed two architectures the one without GAN (SRResNet) and one with GAN (SRGAN). It is concluded that SRGAN has better accuracy and generate image more pleasing to eyes as compared to SRGAN.

**Architecture:**Similar to GAN architectures, the Super Resolution GAN also contains two parts Generator and Discriminator where generator produces some data based on the probability distribution and discriminator tries to guess weather data coming from input dataset or generator.  Generator than tries to optimize the generated data so that it can fool the discriminator. Below are the generator and discriminator architectural details:



**Generator Architecture:**

The generator architecture contains residual network instead of deep convolution networks because residual networks are easy to train and allows them to be substantially deeper in order to generate better results. This is because the residual network used a type of connections called skip connections.



There are B residual blocks (16), originated by ResNet. Within the residual block, two convolutional layers are used, with small 3×3 kernels and 64 feature maps followed by batch-normalization layers and ParametricReLU as the activation function.

The resolution of the input image is increased with two trained sub-pixel convolution layers.

This generator architecture also uses parametric ReLU as an activation function which instead of using a fixed value for a parameter of the rectifier (alpha) like LeakyReLU. It adaptively learns the parameters of rectifier and   improves the accuracy at negligible extra computational cost

  During the training, A high-resolution image (HR) is downsampled to a low-resolution image (LR). The generator architecture than tries to upsample the image from low resolution to super-resolution. After then the image is passed into the discriminator, the discriminator and tries to distinguish between a super-resolution and High-Resolution image and generate the adversarial loss which then backpropagated into the generator architecture.

**Discriminator Architecture:**

The task of the discriminator is to discriminate between real HR images and generated SR images.   The discriminator architecture used in this paper is similar to DC- GAN architecture with LeakyReLU as activation. The network contains eight convolutional layers with of 3×3 filter kernels, increasing by a factor of 2 from 64 to 512 kernels. Strided convolutions are used to reduce the image resolution each time the number of features is doubled. The resulting 512 feature maps are followed by two dense layers and a leakyReLU applied between and a final sigmoid activation function to obtain a probability for sample classification.

