Generative Adversarial Network
(ELL793: COMPUTER VISION)
Prof. Brijesh Lall

Tushar Verma (2022AIY7514)
ScAI, IIT Delhi

1 Introduction

Generative Adversarial Networks (GANs), introduced by Ian Goodfellow and his colleagues in 2014, represent a major breakthrough in the field of deep learning. GANs are a type of generative model that can learn to create new data instances that resemble the training data. They are particularly useful for tasks such as image generation, video generation, and data augmentation. At the core of a GAN is a two-player game between two neural networks: the **generator** and the **discriminator**.

1.1 1. Generator

The generator, G, takes random noise as input and tries to generate data that mimic the real data distribution. The aim of the generator is to "fool" the discriminator by producing outputs that are as realistic as possible.

1.2 2. Discriminator

The discriminator, D, is a binary classifier that takes both real data and the data generated by G as input and attempts to distinguish between the two. The goal of D is to correctly classify whether the input data is real (from the training dataset) or fake (generated by G).

1.3 3. Adversarial Training Process

The two networks are trained in an adversarial manner. The generator improves its ability to generate realistic data by trying to maximize the probability that the discriminator classifies its outputs as real. On the other hand, the discriminator improves its ability to detect fake data by trying to minimize the probability of being fooled by the generator. Formally, GANs can be represented by the following minimax objective function:

$$\min_G \max_D \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))] \tag{1}$$

In this formulation, $p_{data}(x)$ represents the real data distribution, $p_z(z)$ is the noise distribution, $G(z)$ is the data generated by the generator from noise z, and $D(x)$ is the discriminator's estimate of the probability that x is real.

2 Dataset

The CIFAR-10 dataset is a widely used benchmark in machine learning and computer vision. It consists of 60,000 color images, each with a size of 32x32 pixels, divided into 10 different classes. The dataset is split into 50,000 training images and 10,000 test images. Each class contains 6,000 images, and the images are spread evenly across the following 10 classes: *Airplane, Automobile, Bird, Cat, Deer, Dog, Frog, Horse, Ship, Truck.* the CIFAR-10 dataset serves as a fundamental testbed for evaluating a GAN's ability to generate realistic images.

3 Model Architecture

Table 1: Architecture of the Discriminator and Generator

Discriminator	**Input**	**Output**
Conv2d (4x4, stride=2, pad=1)	3 (image channels)	64
Conv2d (4x4, stride=2, pad=1)	64	128
BatchNorm2d + LeakyReLU (0.2)	128	-
Conv2d (4x4, stride=2, pad=1)	128	256
BatchNorm2d + LeakyReLU (0.2)	256	-
Conv2d (4x4, stride=2, pad=0)	256	1
Sigmoid	1	-
Generator	**Input**	**Output**
ConvTranspose2d (4x4, stride=1, pad=0)	100 (latent)	512
ConvTranspose2d (4x4, stride=2, pad=1)	512	256
BatchNorm2d + ReLU	256	-
ConvTranspose2d (4x4, stride=2, pad=1)	256	128
BatchNorm2d + ReLU	128	-
ConvTranspose2d (4x4, stride=2, pad=1)	128	3
Tanh	3	-

A simple 3 layer architecture was sufficient for the task.
Generator: The input latent vector (random noise) of dimension 100 is upsampled through a series of transpose convolution layers. Batch normalization is applied to stabilize training, and ReLU is used as the activation function except in the output layer. The final layer uses a Tanh activation function to produce an output with values between -1 and 1, corresponding to an RGB image.
Discriminator: The input image (3 channels) is downsampled through a series of convolutional layers. LeakyReLU is used as the activation function for all layers except the final one, which uses a sigmoid to produce a binary output (real or fake).

4 Results Through Epochs

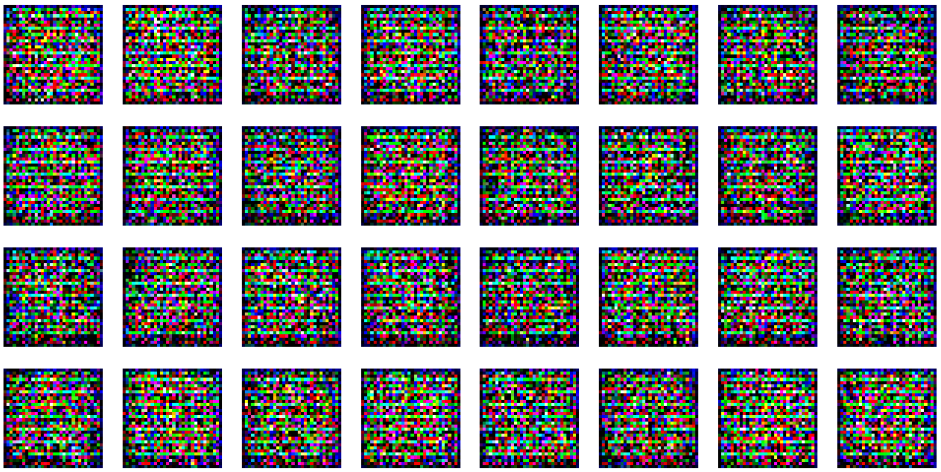

Figure 1: Random Initialization

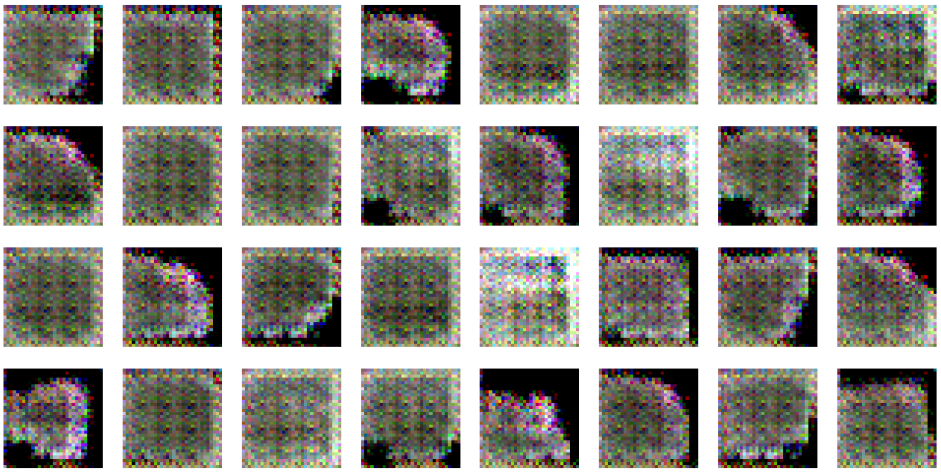

Figure 2: After First Epoch

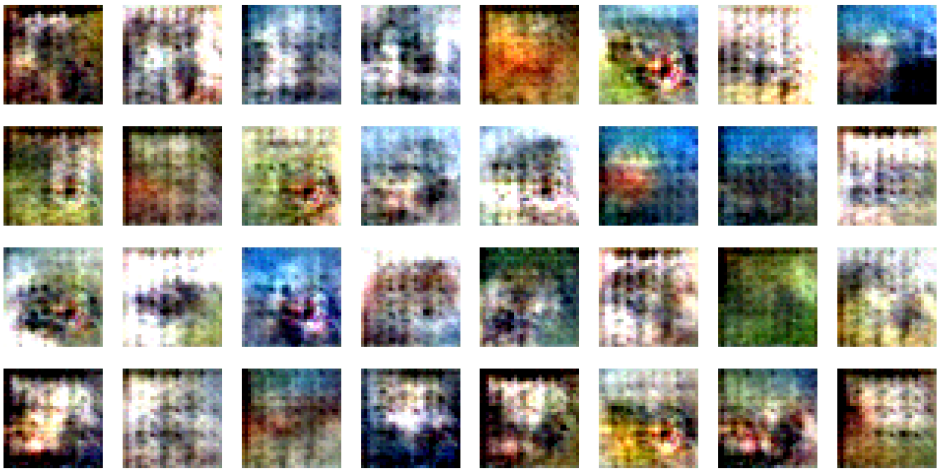

Figure 3: After Second Epoch

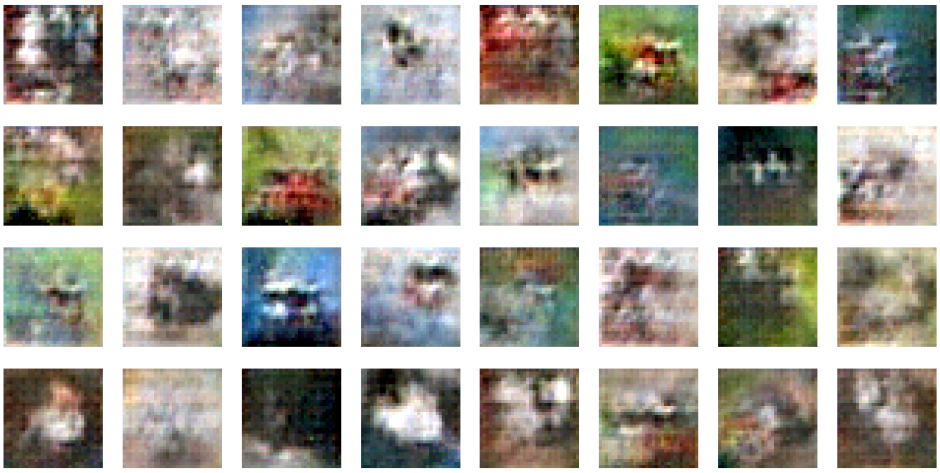

Figure 4: After Fifth Epoch

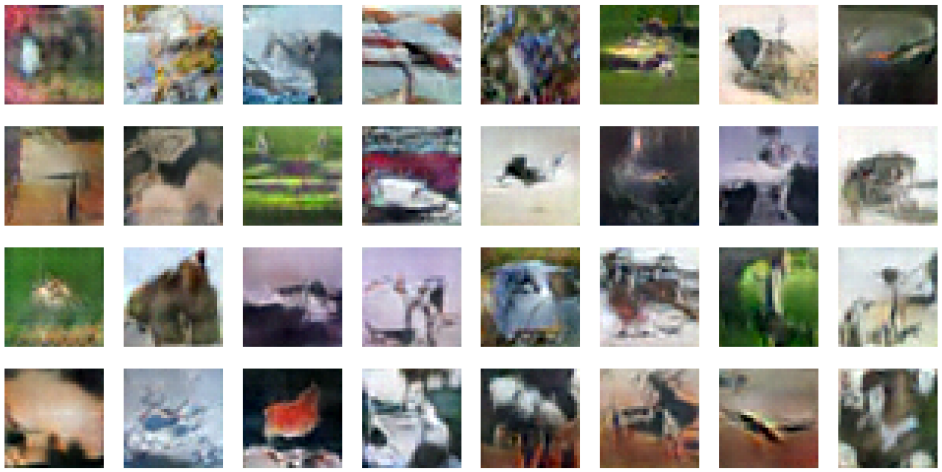

Figure 5: After Twenty Epoch

5 Evaluation

Inception Score (IS)
The Inception Score (IS) is a metric used to evaluate the quality and diversity of images generated by Generative Adversarial Networks (GANs). It aims to measure how realistic and diverse the generated images are, using the pre-trained Inception v3 network, which was trained on the ImageNet dataset. How Inception Score Works:

The score is based on two main concepts:

- **Image Quality**: A good GAN should generate images that can be confidently classified into distinct object categories by the Inception network. If the generated image looks real, the Inception model should give a high probability for a specific class.
- **Image Diversity**: The generated images should cover a wide range of classes. If a GAN only generates images of one or a few classes, it indicates a lack of diversity.

5.1 Inception Score Ranges for GANs

The Inception Score typically ranges from 1 to 10, though it is unbounded in theory. Here's a rough interpretation of the ranges:

- **IS = 1**: The model generates highly random or poor-quality images. All images are classified into many different classes with equal probability, meaning no class dominates.
- **IS between 2 and 4**: This is often considered moderate image quality and diversity. The generated images are somewhat realistic, but may lack diversity or class confidence.

- **IS between 4 and 8**: This range suggests good image quality and diversity. The GAN is generating images that are both realistic and varied.
- **IS grater than 8**: This indicates very high-quality and diverse image generation. The GAN is performing excellently, producing images that the Inception network classifies confidently into distinct classes.

5.2 My Implemented GAN's Inception Score

A score of **3.5** was achieved, suggesting the GAN's generated images have moderate quality and diversity. The images are somewhat realistic, but there may still be room for improvement in:

- **Image sharpness or detail**: The generated images may still look artificial or blurry.
- **Diversity**: The GAN might be generating images from only a subset of the possible classes or generating similar-looking images across classes.

Overall, a score of 3.5 is modest, indicating the GAN is doing reasonably well, considering such a simple architecture.