Homework-4 Report

Anjan Kumar Depuru

Deep learning – CPSC 8430

GitHub link- https://github.com/Anjandepuru/Deep-Learning-HW-4

Introduction-

We build a Generative Adversarial Network (GAN), a kind of unsupervised neural network, for this job. The CIFAR10 dataset will be used to train a pair of discriminator and generator models utilizing ACGAN, Wasserstein GANs, and DCGAN approaches. Creating false photos that closely match the real photographs in the collection is the goal.

GANs, which stand for Generative Adversarial Networks, are computational models that involve two neural networks competing against each other to generate new, synthetic data that can be indistinguishable from real data. They are frequently used in the image, video, and speech generation fields.

WGAN is designed to address some of the challenges with traditional GANs, such as mode collapse and instability during training, by using a Lipschitz continuity constraint on the discriminator's weights.

DCGAN stands for Deep Convolutional Generative Adversarial Networks. It is a type of GAN that uses deep convolutional neural networks (CNNs) as the generator and discriminator models. DCGAN is designed to address some common issues with traditional GANs, such as mode collapse, using architectural constraints and training strategies. DCGAN has been used to generate high-quality images in various applications, such as image synthesis, image-to-image translation, and style transfer.

ACGAN is useful in applications such as image generation with specific attributes or image-to-image translation. ACGAN has been successfully applied to various domains, such as generating realistic images of faces with specific attributes (e.g., age, gender, and expression) and generating different styles of clothing from input sketches.

Problem statement-

To Train a pair of discriminator and generator models on the CIFAR10 dataset using methods derived from DCGAN, Wasserstein GANs, and ACGAN. These techniques involve the use of deep convolutional neural networks, the Wasserstein distance metric, and incorporating a classifier network into the models to generate realistic images with specific attributes. The aim is to generate fake images that closely resemble the real images in the CIFAR10 dataset.

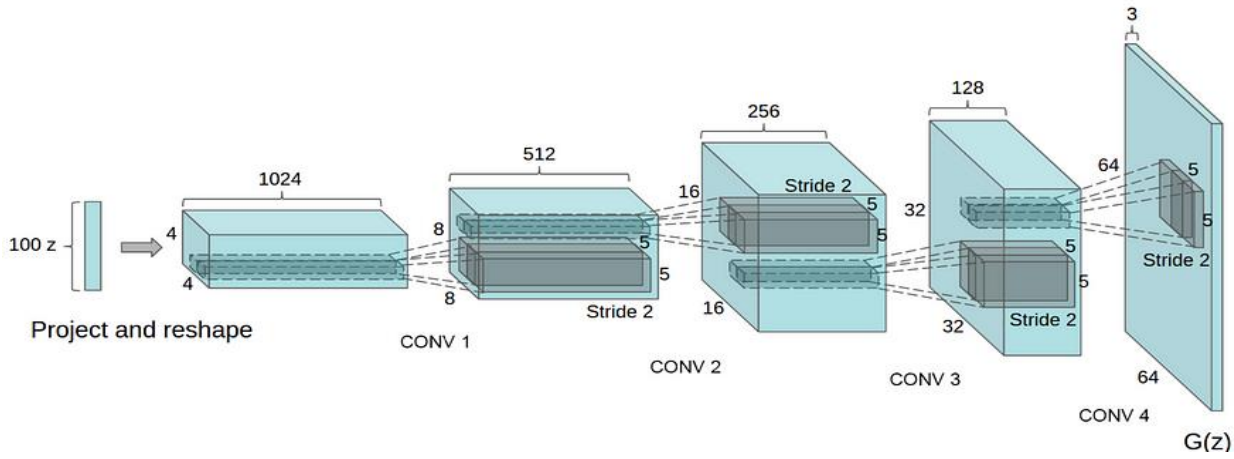
Dataset-

The 60,000 32x32 pixel pictures that make up the CIFAR-10 dataset are all coloured and have a size of 60,000. Airplanes, Cars, Birds, Cats, Deer, Dogs, Frogs, Horses, Ships, Trucks are among the ten categories into which these pictures are broken up. The 6,000 photos in each category. The CIFAR10 dataset is used by all three networks as training data.

'Torch-vision' from Pytorch is used to download the dataset. Cifar-10 package for the dataset. • The created phony photos are of inferior quality because the images in the CIFAR-10 dataset are also subpar.

DCGAN-

The DCGAN network design is one of the most well-liked and effective GAN network architectures. Convolution layers are what it mostly consists of; completely connected or max pooled layers are not included. The network plan for the generator is shown in the image below. This assignment employs batch normalization.



DCGAN employs a unique strategy for down and upsampling using coevolutionary stride and transposed convolutions. This adaptability of DCGAN makes it highly effective and successful in its applications.

To upsample and downsample the images, DCGAN utilizes transposed convolutions and convolutional strides. However, due to the difficulty in training DCGANs, certain architectural criteria need to be met to prevent the model from collapsing.

The network was trained for 50 epochs, and I used the following hyperparameters.

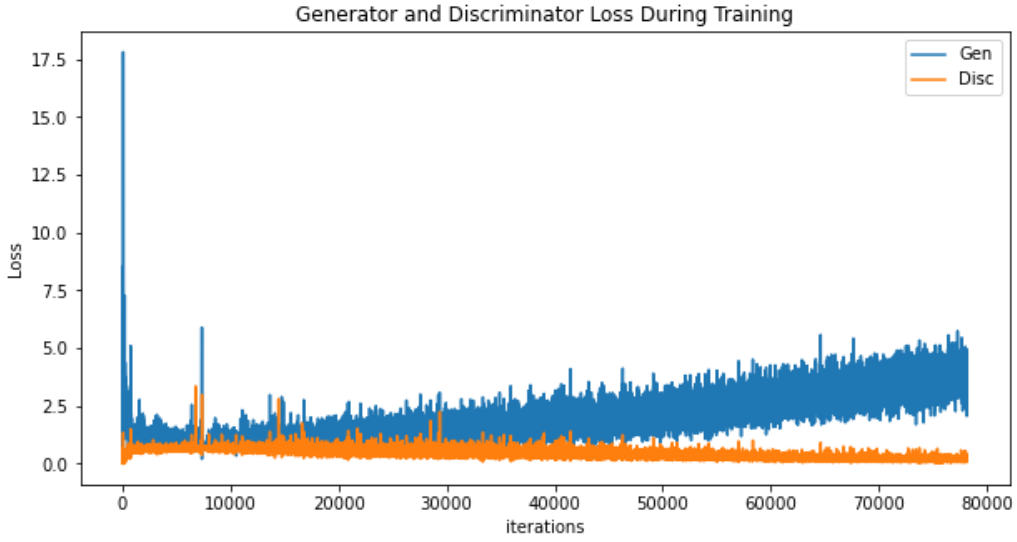
- Learning rate- 2e-4,
- BATCH_SIZE = 128
- NUM_EPOCHS = 50
- Optimizer = Adam
- beta = 0.5
- Momentum = 0.9

- criterion = nn.BCELoss()
- features_disc = 64, this shows the size of the feature map in the discriminator
- features_gen = 64, this shows the size of feature maps in the generator

Figure 2 displays the output images produced by my DCGAN network, which of the three networks was discovered to be the simplest to implement. Additionally, it was discovered that the network was unstable and had varied loss levels between epochs.

FIGURE 2

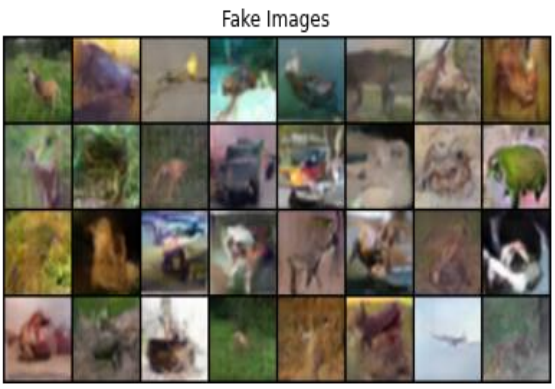
The generator and discriminator losses were as follows 50 epochs, as shown below.



.

We can observe that the generator and discriminator loss settle and hold a constant ratio, like the ideal state, at roughly 10000 iterations.

We can compare Real images and fake images for training models for 50 epochs.



WGAN-

The Wasserstein GAN (Generative Adversarial Network) is an improved version of the standard deep coevolutionary, generative opponent network (DCGAN) used for image generation. It includes a loss feature that focuses on image quality and makes the model training more reliable. Despite requiring only a few minor changes from the standard DCGAN, the WGAN has a strong mathematical foundation.

The WGAN model replaces the sigmoid function in the output layer of the essential generator model with a linear activation feature. Real photographs are represented as -1, and fake pictures are represented as 1, instead of using 0 and 1. The training of both the generator and discriminator models in WGAN is done using Wasserstein distance, which is a mathematical concept used to measure the distance between two probability distributions.

The advantage of using WGAN is that its training process is more reliable and less affected by the model's architecture and selection of hyperparameters. One of the most significant benefits is that the discriminator's loss is more closely related to the quality of the images generated by the generator model.

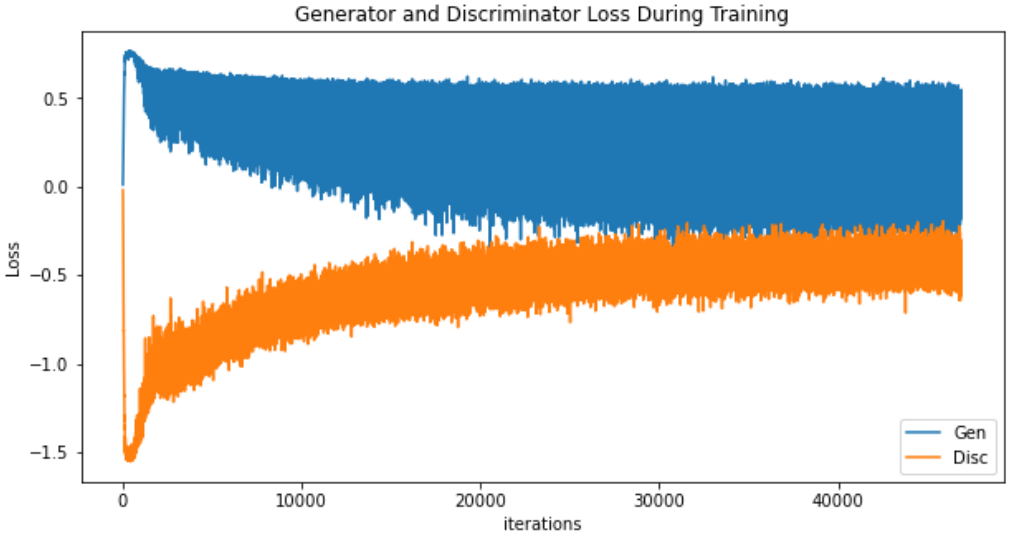
The following are the hyperparameters used to tune this network:

My use of LEARNINGBATCH_SIZE = 32, IMAGE_SIZE = 64, CHANNELS_IMG = 3, NUM_EPOCHS = 30, FEATURES_DISC = 64, FEATURES_GEN = 64, and _RATE = 5e-5; weight_clip = 0.01

The output images generated by my DCGAN network are shown in Figure 3 and Compared to the DCGAN network, the WGAN network was far more reliable. It exhibits the outcome in Figure 3 by following the training images.

Fig 3

Following is the Generator and Discriminator loss generated for 30 epochs:



ACGAN

ACGAN stands for Auxiliary Classifier Generative Adversarial Network. It is a type of generative adversarial network (GAN) that incorporates an additional classifier to the discriminator network**.** The Auxiliary Classifier GAN (AC-GAN) is a type of GAN that extends the Conditional GAN. In this model, instead of taking the class label of an image as input, the discriminator predicts it. This improves the stability of.The training process and allows for the creation of large, high-quality images while also learning a representation in the latent space that is not dependent on the class label Like the Conditional GAN, the AC-GAN's generator model also receives a class label along with a point in the latent space to generate images that are conditional on the specified class

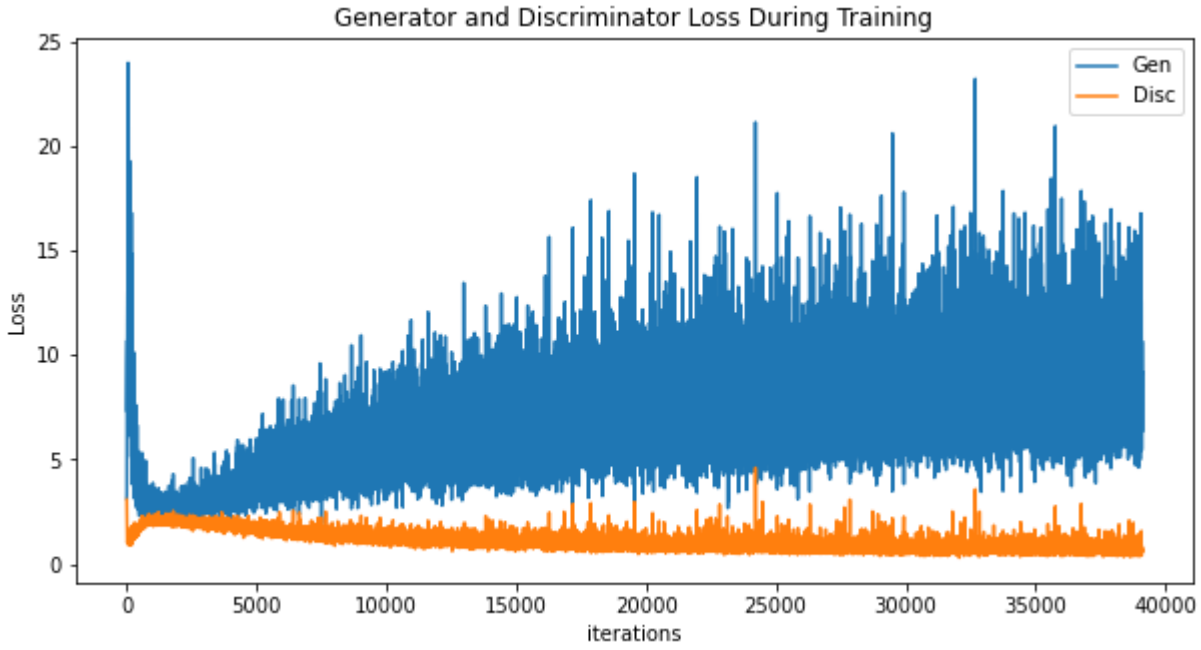
In ACGAN, the generator is trained to generate images that not only look realistic but also belong to specific categories. The discriminator network not only distinguishes between real and fake images but also predicts the category of the image. The additional classifier is trained to predict the category of the image generated by the generator, while the discriminator is trained to distinguish between real and fake images and predict their category**.**

In practice, the discriminator and the auxiliary classifier in a conditional GAN are implemented as two separate models with shared weights. This means that the two models share some of the same internal representations and can learn from each other, while still performing their distinct tasks of distinguishing between real and fake images and predicting the class label of the image, respectively**.**

The following are the hyperparameter used for the network:

The learning rate is 1.5e-4,64 BATCH_SIZE The image size is 64, Channels_Ignition = 3, When NOISE_DIM is 100, NUM_EPOCHS equals 50, a Feature disk size is 64, Features Generation: 64, NUM_CLASSES = 10, 100 EMBED_SIZE

Generator and Discriminator loss



The images generated by the ACGAN model: