

## **EX 10 IMAGE GENERATION USING GENERATIVE ADVERSARIAL**

**DATE: 16/10/2025**

### **Problem Statement:**

Train a Generative Adversarial Network (GAN) to generate new images from a dataset. Evaluate the quality of generated images using both visual inspection and a quantitative metric such as Inception Score (IS) or Fréchet Inception Distance (FID).

Suggested Dataset: LSUN Dataset

### **Objectives:**

1. Understand the architecture of GANs and their adversarial training process.
2. Use StyleGAN2-ADA, a state-of-the-art GAN, for high-quality image generation.
3. Evaluate generated images using IS and FID metrics.
4. Compare generated outputs with real samples and assess realism.

### **Scope:**

GANs are powerful generative models used in image synthesis, super-resolution, style transfer, and more. This experiment familiarizes students with modern GAN architectures and helps develop understanding in both visual and metric-based evaluation of generative quality.

Tools and Libraries Used:

1. Python 3.x
2. PyTorch
3. torchvision
4. StyleGAN2-ADA
5. torchmetrics
6. FID & Inception Score libraries
7. CelebA Dataset

### **Implementation Steps:**

#### **Step 1: Clone StyleGAN2-ADA Repository**

```
REPO = 'stylegan2-ada-pytorch'  
REPO_URL = 'https://github.com/NVlabs/stylegan2-ada-pytorch.git'
```

```

def clone_repo():    if not
os.path.exists(REPO):
subprocess.run(['git', 'clone',
REPO_URL])
sys.path.append(os.path.abspath(RE
PO))

```

## **Step 2: Download Pretrained Generator**

```

MODEL_URL      =      'https://nvlabs-fi-cdn.nvidia.com/stylegan2-
adapytorch/pretrained/ffhq.pkl'
MODEL_PATH = 'ffhq.pkl'

```

```

def download_model():    if
not
os.path.exists(MODEL_PATH):
    import urllib.request
    urllib.request.urlretrieve(MODEL_URL, MODEL_PATH)

```

## **Step 3: Generate Synthetic Images**

```

def generate_images():
    import
    legacy
    import
    dnnlib
    G = legacy.load_network_pkl(open(MODEL_PATH,
'rb'))['G_ema'].to(device)    os.makedirs('generated_images',
exist_ok=True)    imgs = []
    for i in range(1): # NUM_IMAGES = 1
        z = torch.randn([1, G.z_dim],
device=device)    label = torch.zeros([1,
G.c_dim], device=device)
        img = (G(z, label, truncation_psi=0.5,
noise_mode='const') + 1) * 0.5    save_image(img,
f'generated_images/fake_{i}.png')
        imgs.append(img.cpu())
    return torch.cat(imgs)

```

## **Step 4: Calculate Inception Score**

```

from torchmetrics.image.inception import InceptionScore

def calculate_inception_score(imgs):    loader =
DataLoader(TensorDataset((imgs * 255).clamp(0, 255).to(torch.uint8)),
batch_size=32)    is_metric = InceptionScore()    for batch in loader:
        is_metric.update(batch[0])
        mean, std = is_metric.compute()
    print(f'Inception Score: {mean:.2f} ± {std:.2f}')

```

## **Step 5: Prepare Real Images from CelebA**

```
def prepare_real_images():
    dataset = dsets.CelebA(root='./data', split='train',
                           transform=transforms.Compose([
                               transforms.Resize(512),
                               transforms.CenterCrop(512),
                               transforms.ToTensor()
                           ]),
                           download=True)
    loader = DataLoader(dataset, batch_size=1,
                        shuffle=True)  os.makedirs('real_images',
                        exist_ok=True)  for i, (img, _) in
                        enumerate(loader):
        if i >= 1: break
        save_image(img, f'real_images/real_{i}.png')
```

## **Step 6: Compute FID Between Real and Generated Images**

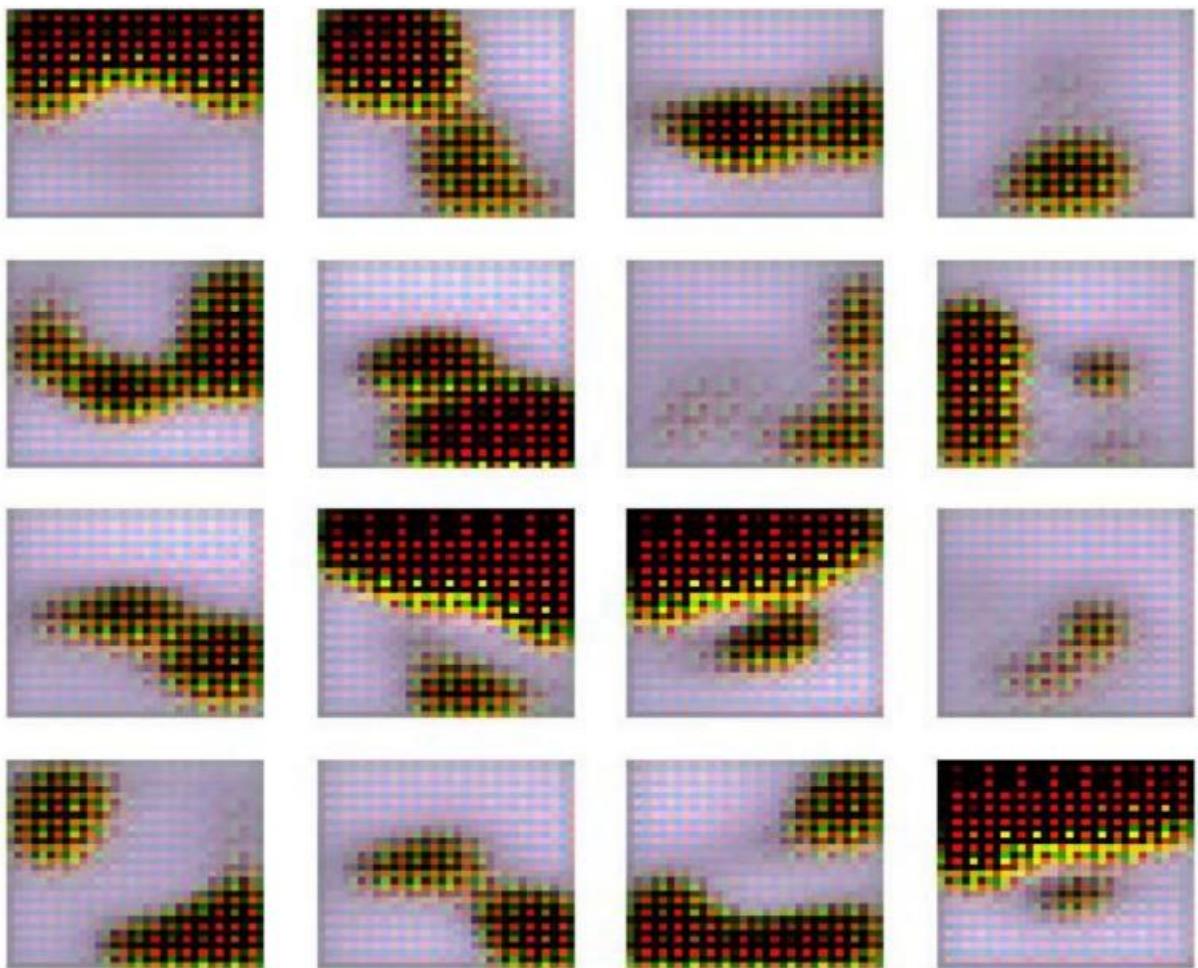
```
def calculate_fid():
    subprocess.run(["python", "-m", "pytorch_fid", "real_images",
                  "generated_images"])
```

## **Step 7: Run Entire Workflow**

```
def main():
    clone_repo()
    download_model()
    imgs =
    generate_images()
    calculate_inception_sco
    re(imgs)
    prepare_real_images()
    calculate_fid()

if __name__ == "__main__":
    main()
```

**OUTPUT:**



**Conclusion:**

This experiment highlights the capabilities of advanced GANs like StyleGAN2 in generating photorealistic images. The use of Inception Score and FID provides quantitative support to visual assessments. GANs remain a cornerstone in generative deep learning research, with wide applications in media, design, and beyond.

