

Cover Sheet

Faculty name : Computers and Artificial Intelligence - Helwan university

Course name : Selected Topics CS-2

Team number 18

Name	ID
محمد سامح احمد	202000763
شهاب جمال الدين السيد	202000442
عبدالرحمن احمد حمدى	202000497
مايا احمد عبد الستار	202000710
محسن هشام محمد	202000715
نانسى احمد مصطفى	202000980

Paper details

Paper name : UNSUPERVISED REPRESENTATION LEARNING WITH DEEP CONVOLUTIONAL GENERATIVE ADVERSARIAL NETWORKS

Authors name: Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun

Publisher name : Soumith Chintala

Content : DC_GAN

Year of publish : 2016

Dataset : The ImageNet

About dataset : The ImageNet dataset is a large-scale image classification dataset that contains over 1.2 million images across 1,000 categories. The dataset is widely used in computer vision research and is often used as a benchmark for evaluating new image classification models.

The implemented algorithms : the ResNet architecture

Results : the authors trained ResNet models with up to 152 layers on the ImageNet dataset and achieved a top-5 error rate of 3.57%, which was significantly better than the previous state-of-the-art result of 3.57%. They also showed that ResNet models can be trained faster and with fewer parameters than previous deep neural network architectures.

General Information on the selected dataset

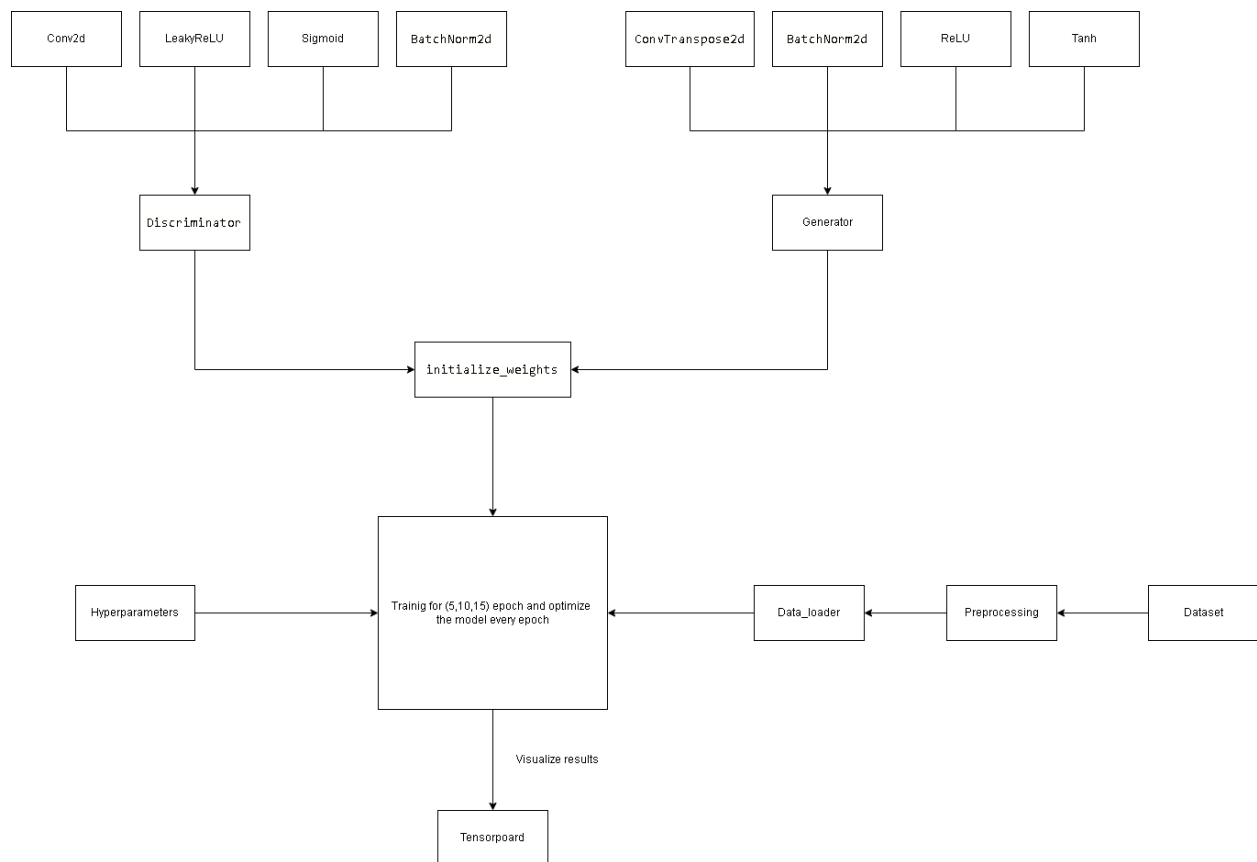
Name : Human Faces

Description : A web scraped dataset of human faces suggested for image processing models

Link : <https://www.kaggle.com/datasets/ashwingupta3012/human-faces>

Total number of samples : 7219

Block diagram for the model



Hyperparameters

Learning rate : $2e-4$ (0.0002)

Batch size : {64, 128, 256}

Image size : 64

Noise Dimensions : 100

Number of epochs : {5, 10, 15}

Discriminator features : 64

Generator features : 64

Optimizer : Adam

Results details

First run

Hyperparameters :

Learning_Rate = $2e-4$

Batch_Size = 128

Image_size = 64

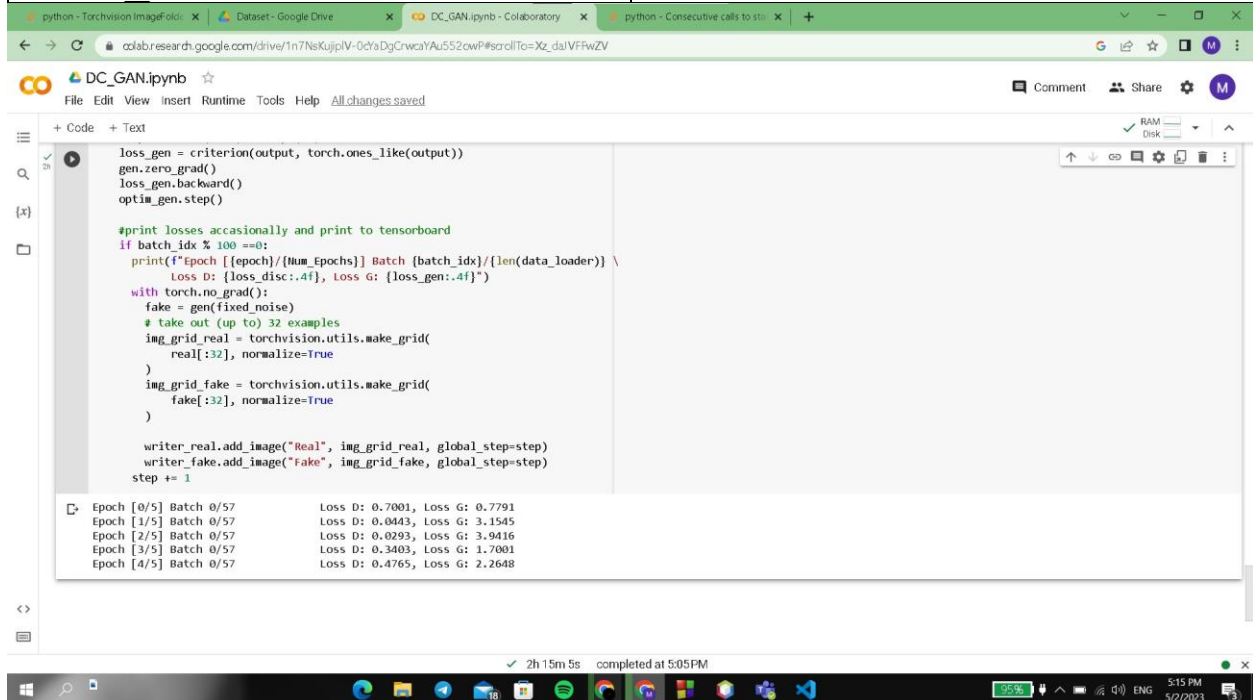
Channels_img = 3

Z_dim = 100

Num_Epochs = 5

CPU

loss_G	2.26
Loss D	0.47



The screenshot shows a Google Colab notebook titled "DC_GAN.ipynb". The code defines a GAN with a generator and discriminator. The training loop prints losses and generates images. The output shows the progress of the first 5 epochs.

```
loss_gen = criterion(output, torch.ones_like(output))
gen.zero_grad()
loss_gen.backward()
optim_gen.step()

#print losses occasionally and print to tensorboard
if batch_idx % 100 == 0:
    print(f'Epoch [{epoch}/{Num_Epochs}] Batch {batch_idx}/{len(data_loader)} \
          Loss D: {loss_disc:.4f}, Loss G: {loss_gen:.4f}')
    with torch.no_grad():
        fake = gen(fixed_noise)
        # take out (up to) 32 examples
        img_grid_real = torchvision.utils.make_grid(
            real[:32], normalize=True
        )
        img_grid_fake = torchvision.utils.make_grid(
            fake[:32], normalize=True
        )

        writer_real.add_image("Real", img_grid_real, global_step=step)
        writer_fake.add_image("Fake", img_grid_fake, global_step=step)
        step += 1
```

Epoch [0/5] Batch 0/57 Loss D: 0.7001, Loss G: 0.7791
Epoch [1/5] Batch 0/57 Loss D: 0.0443, Loss G: 3.1545
Epoch [2/5] Batch 0/57 Loss D: 0.0293, Loss G: 3.9416
Epoch [3/5] Batch 0/57 Loss D: 0.3403, Loss G: 1.7001
Epoch [4/5] Batch 0/57 Loss D: 0.4765, Loss G: 2.2648

2h 15m 5s completed at 5:05PM

Second run

Hyperparameters :

Learning_Rate = $2e-4$

Batch_Size = 128

Image_size = 64

Channels_img = 3

Z_dim = 100

Num_Epochs = 5

GPU

loss G	1.49
Loss D	0.52

```
output = disc(fake).reshape(-1)
loss_gen = criterion(output, torch.ones_like(output))
gen.zero_grad()
loss_gen.backward()
optim_gen.step()

# print losses occasionally and print to tensorboard
if batch_idx % 100 == 0:
    print(f'Epoch [{epoch}/{Num_Epochs}] Batch {batch_idx}/{len(data_loader)} \
          Loss D: {loss_disc:.4f}, Loss G: {loss_gen:.4f}')
    with torch.no_grad():
        fake = gen.fixed_noise()
        # take out (up to) 32 examples
        img_grid_real = torchvision.utils.make_grid(
            real[:32], normalize=True
        )
        img_grid_fake = torchvision.utils.make_grid(
            fake[:32], normalize=True
        )

        writer_real.add_image("Real", img_grid_real, global_step=step)
        writer_fake.add_image("Fake", img_grid_fake, global_step=step)
    step += 1

Epoch [0/5] Batch 0/57      Loss D: 0.6925, Loss G: 0.7917
Epoch [1/5] Batch 0/57      Loss D: 0.0465, Loss G: 3.1504
Epoch [2/5] Batch 0/57      Loss D: 0.3350, Loss G: 0.1055
Epoch [3/5] Batch 0/57      Loss D: 0.3526, Loss G: 1.8797
Epoch [4/5] Batch 0/57      Loss D: 0.5236, Loss G: 1.4973
```

Tensorboard --logdir=/content/logs

4s completed at 7:54PM

python - Torchvision ImageFolder x Colab Notebooks - Google Drive x DC_GAN.ipynb - Colaboratory x

colab.research.google.com/drive/1n7NsKujplV-0dYaDgCrwcaYAu552cwP#scrollTo=wUjOG_7CDVgXS

DC_GAN.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

Files

- drive
- logs
- sample_data

+ Code + Text

fake

Fake

fake

Step 4

Real

real

Step 4

4s completed at 7:54 PM

100% 7:58 PM 5/3/2023

Enable step selection and data table (Scalars only)

Enable Range Selection

Link by step 0

Card Width

SCALARS

Smoothing

0.6

Tooltip sorting method

Alphabetical

Ignore outliers in chart scaling

Partition non-monotonic X axis

HISTOGRAMS

Mode

Offset

Third run

Hyperparameters :

Learning_Rate = $2e-4$

Batch_Size = 256

Image_size = 64

Channels_img = 3

Z_dim = 100

Num_Epochs = 5

GPU

loss_G	3.71
Loss_D	0.049

The screenshot shows a Google Colab notebook titled "DC_GAN.ipynb". The notebook contains a code cell that prints the loss values for the generator (G) and discriminator (D) networks over 5 epochs. The output shows that the loss for G increases from 0.7832 to 3.7122, while the loss for D decreases from 0.6876 to 0.0490. A TensorBoard window is also visible, showing the training progress.

```
if batch_idx % 100 == 0:
    print(f"Epoch [{epoch}/{Num_Epochs}] Batch [{batch_idx}/{len(data_loader)}] \
          Loss D: {loss_disc:.4f}, Loss G: {loss_gen:.4f}")
    with torch.no_grad():
        fake = gen(fixed_noise)
        # take out (up to) 32 examples
        img_grid_real = torchvision.utils.make_grid(
            real[:32], normalize=True
        )
        img_grid_fake = torchvision.utils.make_grid(
            fake[:32], normalize=True
        )
        writer_real.add_image("Real", img_grid_real, global_step=step)
        writer_fake.add_image("Fake", img_grid_fake, global_step=step)
    step += 1
```

Epoch	Batch	Loss D	Loss G
Epoch [0/5]	Batch 0/29	0.6876	0.7832
Epoch [1/5]	Batch 0/29	0.1320	2.1275
Epoch [2/5]	Batch 0/29	0.0455	3.0867
Epoch [3/5]	Batch 0/29	0.0215	3.7958
Epoch [4/5]	Batch 0/29	0.0490	3.7122

TensorBoard --logdir=/content/logs

TensorBoard TIME SERIES IMAGES INACTIVE

Filter runs (regex) Filter tags (regex)

Run 5s completed at 6:58 PM

python - Torchvision ImageFolder x Colab Notebooks - Google Drive x DC_GAN.ipynb - Colaboratory x +

colab.research.google.com/drive/1n7NsKujplV-0dYaDgCrwcaYAu552cwP#scrollTo=iQyrkmpilgC

DC_GAN.ipynb

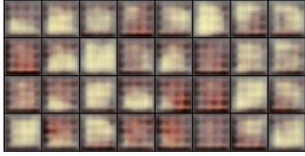
File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

Se [47] Run real fake


Fake

fake Step 4



Real

real Step 0



Link by step 0

Card Width

SCALARS

Smoothing 0.61

Tooltip sorting method Alphabetical

☒ Ignore outliers in chart scaling

☐ Partition non-monotonic X axis

HISTOGRAMS

Mode Offset

IMAGES

Brightness

Contrast

0s completed at 7:02 PM

100% 7:08 PM 5/3/2023

Fourth run

Hyperparameters :

Learning_Rate = $2e-4$

Batch_Size = 128

Image_size = 64

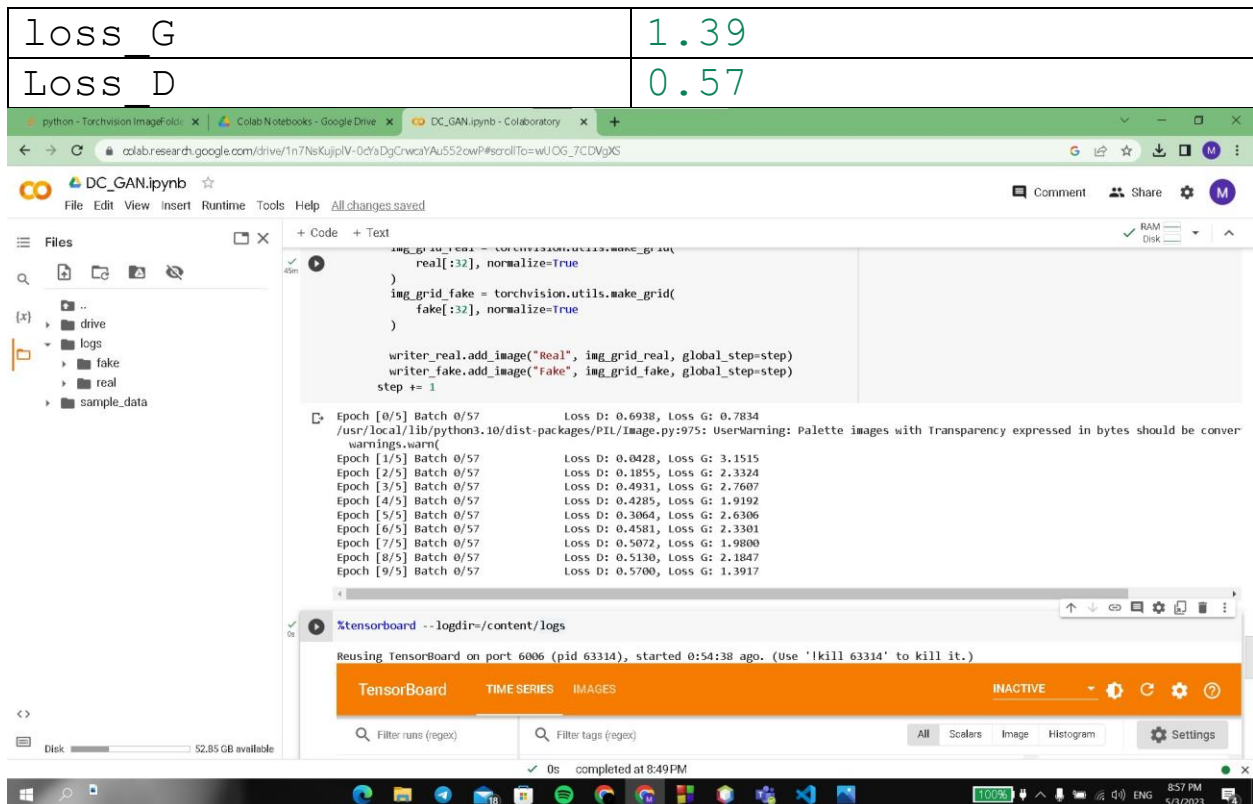
Channels_img = 3

Z_dim = 100

Num_Epochs = 10

GPU

loss_G	1.39
Loss_D	0.57



```
img_grid_real = torchvision.utils.make_grid(
    real[:32], normalize=True
)
img_grid_fake = torchvision.utils.make_grid(
    fake[:32], normalize=True
)

writer_real.add_image("Real", img_grid_real, global_step=step)
writer_fake.add_image("Fake", img_grid_fake, global_step=step)
step += 1
```

Epoch [0/5] Batch 0/57 Loss D: 0.6938, Loss G: 0.7834
/usr/local/lib/python3.10/dist-packages/PIL/Image.py:975: UserWarning: Palette images with Transparency expressed in bytes should be converted to RGBA images
warnings.warn(

Epoch [1/5] Batch 0/57 Loss D: 0.0428, Loss G: 3.1515
Epoch [2/5] Batch 0/57 Loss D: 0.1855, Loss G: 2.3324
Epoch [3/5] Batch 0/57 Loss D: 0.4931, Loss G: 2.7607
Epoch [4/5] Batch 0/57 Loss D: 0.4285, Loss G: 1.9192
Epoch [5/5] Batch 0/57 Loss D: 0.3064, Loss G: 2.6306
Epoch [6/5] Batch 0/57 Loss D: 0.4581, Loss G: 2.3301
Epoch [7/5] Batch 0/57 Loss D: 0.5072, Loss G: 1.9800
Epoch [8/5] Batch 0/57 Loss D: 0.5130, Loss G: 2.1847
Epoch [9/5] Batch 0/57 Loss D: 0.5700, Loss G: 1.3917

TensorBoard --logdir=/content/logs
Reusing TensorBoard on port 6006 (pid 63314), started 0:54:38 ago. (Use 'kill 63314' to kill it.)

TensorBoard TIME SERIES IMAGES INACTIVE Filter runs (regex) Filter tags (regex) All Scalers Image Histogram Settings

python - Torchvision ImageFolder x Colab Notebooks - Google Drive x DC_GAN.ipynb - Colaboratory x +

colab.research.google.com/drive/1n7NsKujplV-0dYaDgCrwcaYAu552cwP#scrollTo=wUjOG_7CDVgXS

DC_GAN.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Settings

Files

- drive
- logs
- fake
- real
- sample_data

Filter runs (regex)

- Run
- real
- fake

Filter tags (regex)

Fake

Fake

Step 9

Real

Real

Step 9

Settings

GENERAL

Horizontal Axis

Step

☐ Enable step selection and data table (Scalars only)

☐ Enable Range Selection

☐ Link by step 0

Card Width

SCALARS

Smoothing

0.6

Tooltip sorting method

Alphabetical

☒ Ignore outliers in chart scaling

☐ Partition non-monotonic X axis

0s completed at 8:49 PM

52.85 GB available

100%

9:00 PM 5/3/2023

Fifth run

Hyperparameters :

Learning_Rate = $2e-4$
Batch_Size = 64
Image_size = 64
Channels_img = 3
Z_dim = 100
Num_Epochs = 10

GPU

loss_G	1.88
Loss_D	0.35

```
writer_real.add_image("Real", img_grid_real, global_step=step)
writer_fake.add_image("Fake", img_grid_fake, global_step=step)
step += 1
```

Epoch [0/5] Batch 0/114 Loss D: 0.6914, Loss G: 0.8111
/usr/local/lib/python3.10/dist-packages/PIL/Image.py:975: UserWarning: Palette images with Transparency expressed in bytes should be converted to RGBA images
Epoch [0/5] Batch 100/114 Loss D: 0.1026, Loss G: 3.2447
Epoch [1/5] Batch 0/114 Loss D: 0.0425, Loss G: 3.9127
Epoch [1/5] Batch 100/114 Loss D: 0.5688, Loss G: 0.6308
Epoch [2/5] Batch 0/114 Loss D: 0.7099, Loss G: 2.4582
Epoch [2/5] Batch 100/114 Loss D: 0.5528, Loss G: 2.8797
Epoch [3/5] Batch 0/114 Loss D: 0.6708, Loss G: 2.1930
Epoch [3/5] Batch 100/114 Loss D: 0.4952, Loss G: 1.6539
Epoch [4/5] Batch 0/114 Loss D: 0.5055, Loss G: 2.0066
Epoch [4/5] Batch 100/114 Loss D: 0.5959, Loss G: 1.6827
Epoch [5/5] Batch 0/114 Loss D: 0.5603, Loss G: 1.5031
Epoch [5/5] Batch 100/114 Loss D: 0.6074, Loss G: 1.5094
Epoch [6/5] Batch 0/114 Loss D: 0.4651, Loss G: 1.5391
Epoch [6/5] Batch 100/114 Loss D: 0.5820, Loss G: 1.8433
Epoch [7/5] Batch 0/114 Loss D: 0.7602, Loss G: 2.9103
Epoch [7/5] Batch 100/114 Loss D: 0.5106, Loss G: 2.5754
Epoch [8/5] Batch 0/114 Loss D: 0.5467, Loss G: 1.8789
Epoch [8/5] Batch 100/114 Loss D: 0.5164, Loss G: 2.2615
Epoch [9/5] Batch 0/114 Loss D: 0.5184, Loss G: 2.3043
Epoch [9/5] Batch 100/114 Loss D: 0.3515, Loss G: 1.8862

TensorBoard --logdir=/content/logs

TensorBoard TIME SERIES IMAGES INACTIVE

