MAY 26, 2025

Deep Convolutional Generative Adversarial Network (DCGAN)

by

MOHAMED KHALED

Abstract: This report presents the implementation and analysis of a Generative Adversarial Network (GAN) trained to generate synthetic image samples. The GAN consists of two primary components: a generator network responsible for producing realistic images from random noise, and a discriminator network tasked with distinguishing between real and generated images. The architectures of both networks are described in detail, highlighting the use of convolutional and transposed convolutional layers for image generation and classification. During training, adversarial loss functions guide the improvement of both networks through a competitive learning process. The report includes the loss history plot to illustrate the training dynamics and convergence behavior. Additionally, a set of ten sample images generated by the trained generator is provided to visually assess the model's performance. The results demonstrate the GAN's ability to produce coherent and realistic image outputs, validating the effectiveness of the adversarial training approach.

1 The generator architecture

The generator network is designed to progressively transform a 1D latent noise vector into a realistic 64x64 RGB image through a series of dense, normalization, activation, reshaping, and transposed convolutional layers. The detailed architecture is as follows:

- Dense Layer: The input noise vector is first passed through a dense layer with 16,384 units, producing a high-dimensional feature vector. This layer has 1,638,400 parameters.
- Batch Normalization: Applied to stabilize training and normalize the outputs.
- LeakyReLU Activation: Introduces non-linearity and helps prevent dead neurons.
- Reshape Layer: The vector is reshaped into an 8×8×256 feature map, preparing it for convolutional operations.
- Conv2DTranspose Layers:
  - First transposed convolution upsamples the feature map to 16×16 with 128 filters (kernel size implied), totaling 819,200 parameters, followed by batch normalization and LeakyReLU.
  - The second transposed convolution upsamples to 32×32 with 64 filters, with 204,800 parameters, batch normalization, and LeakyReLU activation.
  - The final Conv2DTranspose upsamples to the output image size of 64×64 with 3 filters (RGB channels), using 4,800 parameters.
- Total Parameters: The generator contains approximately 2.7 million parameters (2,733,504 total), with 2,700,352 trainable parameters and 33,152 non-trainable parameters.

This architecture enables the generator to learn and produce high-quality images from random noise through progressive upsampling and feature refinement.

```
Layer (type)            Output Shape          Param #
=================================================================
dense_28 (Dense)          (None, 16384)         1,638,400
batch_normalization_45 (Batch (None, 16384)         65,536
Normalization)
leaky_re_lu_71 (LeakyReLU)  (None, 16384)         0
reshape_15 (Reshape)        (None, 8, 8, 256)      0
conv2d_transpose_45 (Conv2DTranspose) (None, 16, 16, 128) 819,200
batch_normalization_46 (BatchNormalization) (None, 16, 16, 128) 512
leaky_re_lu_72 (LeakyReLU)  (None, 16, 16, 128)    0
conv2d_transpose_46 (Conv2DTranspose) (None, 32, 32, 64)  204,800
batch_normalization_47 (BatchNormalization) (None, 32, 32, 64) 256
leaky_re_lu_73 (LeakyReLU)  (None, 32, 32, 64)     0
conv2d_transpose_47 (Conv2DTranspose) (None, 64, 64, 3)    4,800
=================================================================
Total params: 2,733,504
Trainable params: 2,700,352
Non-trainable params: 33,152
```

2 The discriminator architecture

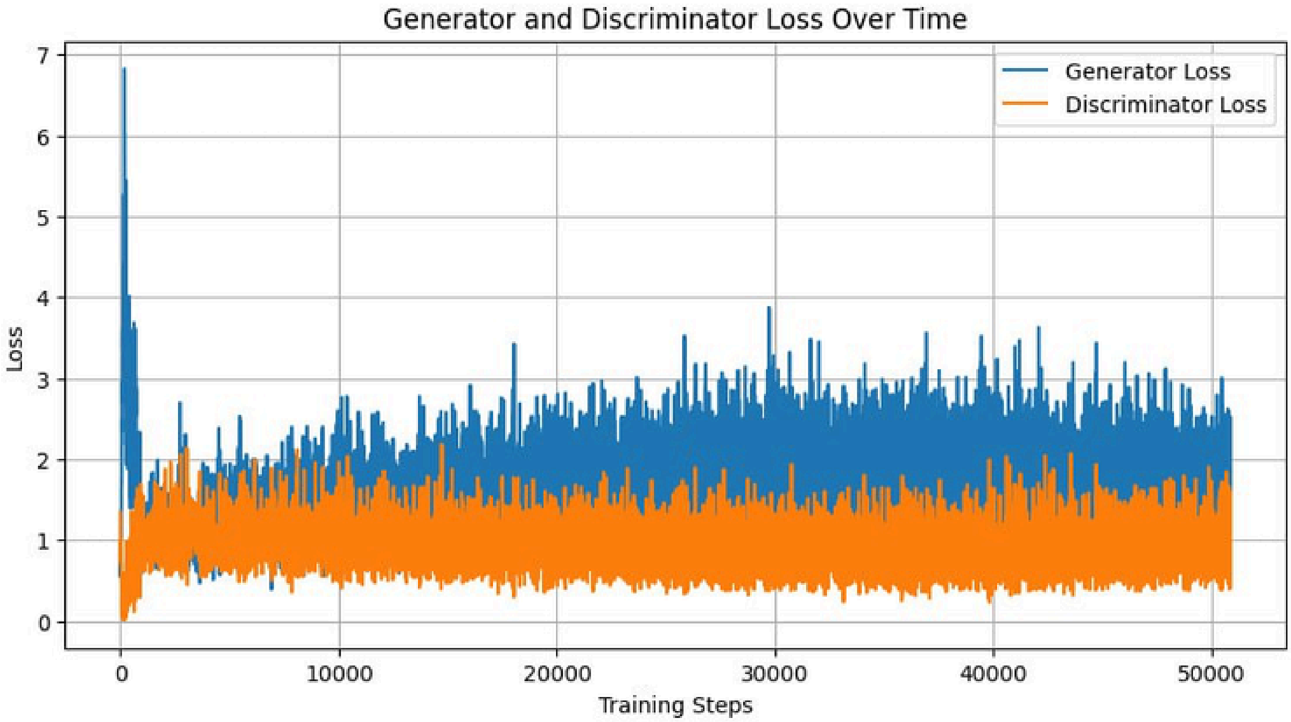
The discriminator network is designed to classify input images as real or fake by progressively extracting features through convolutional layers followed by nonlinear activations and regularization. The architecture details are as follows:

- Conv2D Layer: The input image is processed by a convolutional layer with 64 filters, producing a feature map of size 32×32×64. This layer contains 4,864 parameters.
- LeakyReLU Activation: Applied to introduce non-linearity and allow small gradients when units are not active.
- Dropout Layer: Used to reduce overfitting by randomly dropping units during training.
- Conv2D Layer: A second convolutional layer with 128 filters downsamples the feature map to 16×16×128, containing 204,928 parameters.
- LeakyReLU Activation: Further non-linear transformation.
- Dropout Layer: Additional regularization.
- Flatten Layer: The multidimensional feature map is flattened into a 1D vector of size 32,768 to feed the dense layer.
- Dense Layer: The final layer outputs a single value indicating the probability of the input being real or fake, with 32,769 parameters.
- Total Parameters: The discriminator has approximately 242,561 trainable parameters.

This architecture allows the discriminator to effectively learn discriminative features to differentiate between real and generated images

```
Layer (type)            Output Shape        Param #
=================================================================
conv2d_27 (Conv2D)         (None, 32, 32, 64)    4,864
leaky_re_lu_74 (LeakyReLU)   (None, 32, 32, 64)    0
dropout_26 (Dropout)       (None, 32, 32, 64)    0
conv2d_28 (Conv2D)         (None, 16, 16, 128)   204,928
leaky_re_lu_75 (LeakyReLU)   (None, 16, 16, 128)   0
dropout_27 (Dropout)       (None, 16, 16, 128)   0
flatten_13 (Flatten)       (None, 32768)       0
dense_29 (Dense)          (None, 1)          32,769
=================================================================
Total params: 242,561
Trainable params: 242,561
Non-trainable params: 0
```

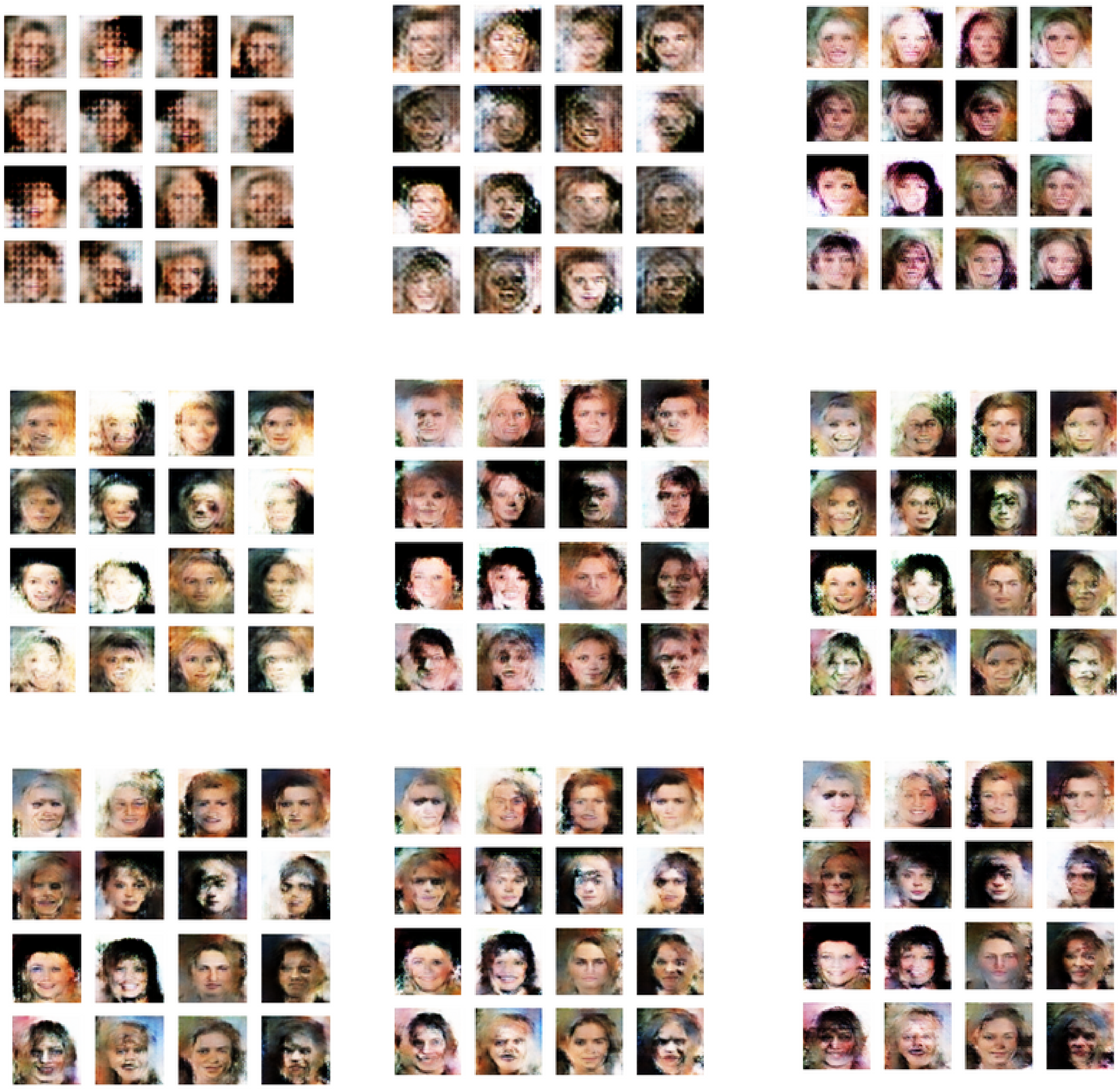
Loss history



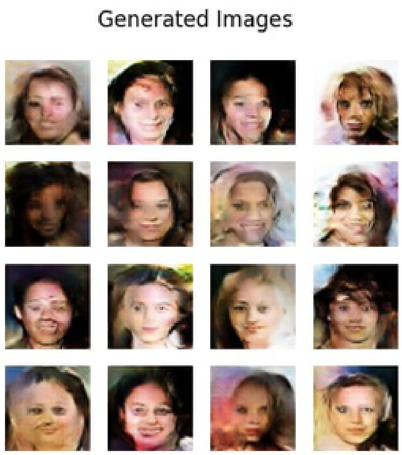
FINAL generator loss:1.4066033

FINAL discriminator loss:0.91312987

10 different sample images generated by the trained generator network.



FINAL Result:
(20 epochs)


Generated Images