Generation of Human Faces

Author: Tommaso Ancilli

DIISM

Preprocessing of Dataset

- Cleaning of the images:
  - I remove noise, being present on edges of the image, by cropping it. Going from a resolution of $216x180$ to a $140x140$
  - Down-sampling the image, going from $140x140$ to a $64x64$
- Feature scaling:
  - None
- Feature selection:
  - Randomly, each image was chosen to be inside the final dataset with a p probability

Discriminator Architecture

```
1 -> Conv2d(3, 32, kernel_size=(7, 7), stride=(1, 1), padding=same)
2 -> AvgPool2d(kernel_size=2, stride=2, padding=0)
3 -> LeakyReLU(negative_slope=0.01, inplace=True)

4 -> Conv2d(32, 64, kernel_size=(5, 5), stride=(1, 1), padding=same)
5 -> AvgPool2d(kernel_size=2, stride=2, padding=0)
6 -> LeakyReLU(negative_slope=0.01, inplace=True)

7 -> Conv2d(64, 128, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
8 -> ReLU(inplace=True)

9 -> Conv2d(128, 256, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
10 -> ReLU(inplace=True)

11 -> Flatten(start_dim=1, end_dim=-1)
12 -> Linear(in_features=4096, out_features=1, bias=True)
13 -> Sigmoid()
```

Classification task: Sigmoid function for output layer

Downsamling the tensor

- Average Pooling
- Zero-padding convolution

Discriminator loss function

Cost function:

- $-\frac{1}{n} * \sum_{i=1}^{n} (y_i \log(D(x)) + (1 - y_i) \log(1 - D(G(z)))$

Gradient-base optimizer: ADAM

- Learning rate : $1x10^{-5}$; $\beta_1 = 0.9$; $\beta_2 = 0.999$
- Minimization of the BCE

Generator Architecture

Figure: "Paper-based" Generative architecture

```
1 -> ConvTranspose2d(5, 128, kernel_size=(4, 4), stride=(1, 1), padding=(2, 2))
2 -> ReLU()

3 -> BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
4 -> ConvTranspose2d(128, 64, kernel_size=(3, 3), stride=(3, 3), padding=(2, 2))
5 -> ReLU()

6 -> BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
7 -> ConvTranspose2d(64, 32, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
8 -> ReLU()

9 -> BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
10 -> ConvTranspose2d(32, 16, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
11 -> ReLU()

12 -> ConvTranspose2d(16, 3, kernel_size=(2, 2), stride=(2, 2))
13 -> ReLU()
```

Figure: Autoencoder for Generative architecture

```
1 -> Conv2d(3, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
2 -> LeakyReLU(negative_slope=0.01, inplace=True)

3 -> Conv2d(64, 128, kernel_size=(5, 5), stride=(1, 1), padding=(2, 2))
4 -> LeakyReLU(negative_slope=0.01, inplace=True)

5 -> Conv2d(128, 256, kernel_size=(7, 7), stride=(1, 1), padding=(3, 3))
6 -> LeakyReLU(negative_slope=0.01, inplace=True)

7 -> Conv2d(256, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
8 -> ReLU(inplace=True)

9 -> Conv2d(64, 3, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
10 -> ReLU(inplace=True)
```

Problem faced

Auto-encoder has too many parameters and with the same learning rate for Discriminator and Generator the former was better to distinguish between real/fake images. Therefore, the min-max game ends up with the perfect separation of real from make-up images by the discriminator, preventing the weights update.
I have used two different solutions::
Changed architecture, I have found a ”sweet-spot” between the ratio of the two learning rate following this paper (here). Concerning the objective function, reading the original paper (here)

Solution for the Generator problem

1 attempt:

- $-\frac{1}{L} * \sum_{i=1}^{L} (y_i \log(D(x)) + (1 - y_i) \log(1 - D(G(z))) = -\frac{1}{L} * \sum_{i=1}^{L} (1 - y_i) \log(1 - D(G(z)))$
- Gradient-base optimizer: ADAM; Learning rate : $1x10^{-3}$; $\beta_1 = 0.9$; $\beta_2 = 0.999$
- Maximization of the objective

2 attempt:

- $-\frac{1}{L} * \sum_{i=1}^{L} (y_i \log(D(x)) + (1 - y_i) \log(1 - D(G(z))) = -\frac{1}{L} * \sum_{i=1}^{L} y_i \log(D(G(z)))$
- Gradient-base optimizer: ADAM; Learning rate : $1x10^{-3}$; $\beta_1 = 0.9$; $\beta_2 = 0.999$
- Minimization of the objective

Results for conventional architecture

Figure: Modified BCE

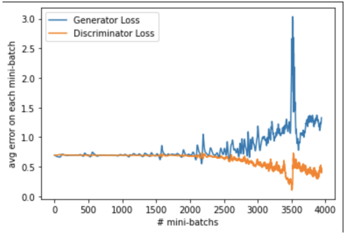
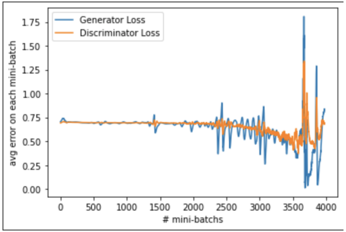


Figure: classic min-max problem with BCE



Results for auto-encoder architecture

Figure: Modified BCE

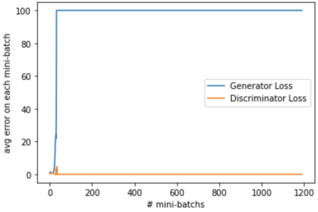
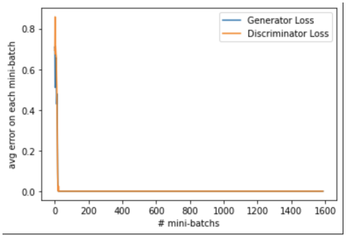
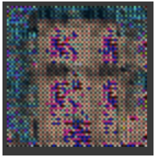
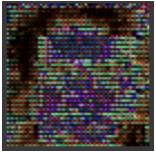


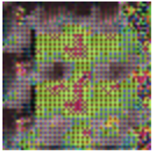
Figure: classic min-max problem with BCE.(shortened to 4 epochs)



Generated faces







Conclusion

- Training process too unstable and the generation of the images is not straightforward
- Limited statistical variability in the image generation procedure