

[◀ Return to "Deep Learning" in the classroom](#)[DISCUSS ON STUDENT HUB](#)

Generate Faces

REVIEW

CODE REVIEW

HISTORY

Meets Specifications

Awesome submission, you have correctly implemented the DCGAN architecture and have used all the hyperparameters correctly. Great job at taking care of all the past reviews. The results look great! Congratulations on passing this project and all the best 🍷

I suggest you read following articles to know how to improve results in GAN:

<https://github.com/soumith/ganhacks#how-to-train-a-gan-tips-and-tricks-to-make-gans-work>

<http://blog.otoro.net/2016/04/01/generating-large-images-from-latent-vectors/>

Required Files and Tests

The project submission contains the project notebook, called "dlnd_face_generation.ipynb".

All the unit tests in project have passed.

Data Loading and Processing

The function `get_data_loader` should transform image data into resized, Tensor image types and return a `DataLoader` that batches all the training data into an appropriate size.

Pre-process the images by creating a `scale` function that scales images into a given pixel range. This function should be used later, in the training loop.

Build the Adversarial Networks

The Discriminator class is implemented correctly; it outputs one value that will determine whether an image is real or fake.

The Generator class is implemented correctly; it outputs an image of the same shape as the processed training data.

This function should initialize the weights of any convolutional or linear layer with weights taken from a normal distribution with a mean = 0 and standard deviation = 0.02.

Optimization Strategy

The loss functions take in the outputs from a discriminator and return the real or fake loss.

There are optimizers for updating the weights of the discriminator and generator. These optimizers should have appropriate hyperparameters.

Training and Results

Real training images should be scaled appropriately. The training loop should alternate between training the discriminator and generator networks.

There is not an exact answer here, but the models should be deep enough to recognize facial features and the optimizers should have parameters that help with model convergence.

The project generates realistic faces. It should be obvious that generated sample images look like faces.

The question about model improvement is answered.

 [DOWNLOAD PROJECT](#)

[RETURN TO PATH](#)

Rate this review