UDACITY

< Return to "Deep Learning" in the classroom

DISCUSS ON STUDENT HUB

Generate Faces

REVIEW

CODE REVIEW

HISTORY

Meets Specifications

Congratulations on completing the project 🎉
Overall, it was a very satisfactory submission 👏🏻

Have fun learning ☺️

Required Files and Tests

The project submission contains the project notebook, called “dlnd_face_generation.ipynb”.

All the unit tests in project have passed.

All unit tests passed 👍🏽

Data Loading and Processing

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

The `get_dataloader` function was properly implemented.
All images were properly loaded and resized to size 32.

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.

The `scale` function scales images into a pixel range of -1 to 1.

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.

Both the Discriminator and Generator classes were implemented correctly.
You can have a bigger model for the generator as it has to perform a much more complex task than the Discrminator.

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.

The `weights_init_normal` function was implemented correctly and used to initialize the weights of both geenrator and discriminator.

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.

```
lr = 0.0002
beta1=0.5
beta2=0.999
```

Good choice of 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 wth model convergence.

Good choice of hyperparameters 👍

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

The generated images are realistic. Great work 👏

The question about model improvement is answered.

Well answered!

DOWNLOAD PROJECT

RETURN TO PATH

Rate this project