UDACITY

PROJECT

Generate Faces

A part of the Deep Learning Nanodegree Foundation Program

PROJECT REVIEW

CODE REVIEW

NOTES

SHARE YOUR ACCOMPLISHMENT!

Meets Specifications

This is an awesome submission. The faces generated are quite clear and realistic. I have left my views, suggestions and comments below to your reference.

In order to get good performance with image generation, there are several different implementations and tricks, which you can check out from https://github.com/soumith/ganhacks

Here are some other important resources for GAN:

http://www.araya.org/archives/1183 for GAN stability.
https://github.com/yihui-he/GAN-MNIST, https://github.com/carpedm20/DCGAN-tensorflow for DCGAN.
https://medium.com/@ageitgey/abusing-generative-adversarial-networks-to-make-8-bit-pixel-art-e45d9b96cee7
It can be seen that a lot of hard work has been put into this.

Congratulations on successfully completing Project 5 and the deep learning nanodegree! :)

Required Files and Tests

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

All the unit tests in project have passed.

Build the Neural Network

The function model_inputs is implemented correctly.

The function discriminator is implemented correctly.

Good work using Batch Normalization.
Nice work using Leaky RELUs, which allow a small, non-zero gradient when the unit is not active.
Try using different values of alpha between 0.08 and 0.15 and compare your results.

The function generator is implemented correctly.

Good work using Batch Normalization.
Nice work using Leaky RELUs which allow a small, non-zero gradient when the unit is not active.
Since tanh is the last layer of the generator output, normalizing the input images is a good step.

You should use different values of alpha between 0.08 and 0.15 and compare your results.
Brilliant Work!!

The function model_loss is implemented correctly.

Good work using smoothing as it prevents discriminator from being from being too strong and to generalize in a better way better way

The function model_opt is implemented correctly.

Good Implementation!!

You have wrapped the training step with dependency to tf.GraphKeys.UPDATE_OPS, which is brilliant.

Neural Network Training

The function train is implemented correctly.

- It should build the model using `model_inputs`, `model_loss`, and `model_opt`.
- It should show output of the `generator` using the `show_generator_output` function

Good job keeping batch_z between -1 and 1.
Good work increasing the batch size by a factor of 2, i.e., batch_images = batch_images * 2, inside the inner for loop.

The parameters are set reasonable numbers.

Good work adjusting the hyper parameters.
Try using different values of learning rate between 0.0002 and 0.0008 and different values of Beta1 between 0.2 and 0.5 and compare your results.

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

The faces generated are quite clear and realistic. :)

DOWNLOAD PROJECT

RETURN TO PATH