Meets Specifications

Awesome submission! 🎉 It was a pleasure reviewing your project.

You've done a great job of implementing a Deep Convolutional GAN. Your results show that you have a great understanding of how GANs work in theory and practice.

If you are curious/want to learn more, I recommend you to take a look at the links mentioned below:
http://blog.otoro.net/2016/04/01/generating-large-images-from-latent-vectors/
https://github.com/soumith/ganhacks#how-to-train-a-gan-tips-and-tricks-to-make-gans-work

Keep learning! ✌️

Required Files and Tests

✓

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

All the necessary files are included 👍

✓

All the unit tests in project have passed.

Good job, all the 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.

✓

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.

Excellent job at

- Considering variable scoping
- Using a sequence of convolutional layers using `conv2d` with strides to avoid making sparse gradients instead of max-pooling layers as they make the model unstable.
- Using `leaky_relu` and avoiding ReLU for the same reason of avoiding sparse gradients as leaky_relu allows gradients to flow backwards unimpeded.
- Using `sigmoid` as output layer.

- Using `sigmoid` as output layer.
- BatchNorm to avoid "internal covariate shift" as batch normalisation minimises the effect of weights and parameters in successive forward and back pass on the initial data normalisation done to make the data comparable across features.
  Awesome. 👌

Some suggestions for improvement:

- Use Dropouts in discriminator so as to make it more robust.

✓

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

Same as the discriminator function: Excellent job at

- Implementing generator as "deconvolution" network with mostly the same key points as mentioned above.
- `Tanh` as the last layer of the generator output. This means that we'll have to normalise the input images to be between -1 and 1.

Suggestions for improvement:

- It is recommended to have a larger generator network than the discriminator network.
- Similar to the discriminator, use Dropouts n the generator at both train and test time with `keep_probability` as 0.5 as suggested here.

✓

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.

Excellent job implementing the loss of the model!

Suggestion for improvement:

- Consider one-sided label smoothing. It's done to prevent discriminator from being too strong as well as to help it generalise better by reducing labels from 1 to 0.9.

✓

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 wth model convergence.

✓

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

Excellent job, you can see how your generator has created faces good enough to fool the discriminator (do they fool you? would you tell them apart from real faces?). How cool is that?

✓

The question about model improvement is answered.

DOWNLOAD PROJECT