UDACITY Logout

PROJECT

Generate Faces

A part of the Deep Learning Nanodegree Foundation Program

PROJECT REVIEW | CODE REVIEW | NOTES

Requires Changes

SHARE YOUR ACCOMPLISHMENT

1 SPECIFICATION REQUIRES CHANGES

Great first submission! You are almost there. You just need to normalize the batch_images. If you would like to improve the result further, please consider applying other suggestions. I have added some notes for you to look at.

Keep up the hard work! Looking forward to your next submission.

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 the unit tests in project have passed. Great job! 👏

Build the Neural Network

✓ **The function model_inputs is implemented correctly.**

✓ **The function discriminator is implemented correctly.**

Excellent job with:

- Great choice of architecture
- Using sequence of `conv2d` layers with strides
- Using LeakyReLu.
- Applying batch normalization with `tf.layers.batch_normalization` to normalize the convolutional layer's output on each layers except the first convolutional layer and the output layer.

Suggestion:

- Initializing weight by passing `kernel_initializer=tf.contrib.layers.xavier_initializer()` in `tf.layers.conv2d`. This initializer is designed to keep the scale of the gradients roughly the same in all layers. Here is the resource https://www.tensorflow.org/api_docs/python/tf/contrib/layers/xavier_initializer
- 0.2 alpha is good, however trying to lower it to 0.1.
- Adding a dropout with high keep_probability is simple way to help the network generates better results. For example, `tf.nn.dropout(x3, 0.8)`. Here is the resource: https://www.tensorflow.org/api_docs/python/tf/nn/dropout

✓ **The function generator is implemented correctly.**

Suggestion:
Simplify `reuse=False if is_train==True else True` as `reuse = not is_train`

✓ **The function model_loss is implemented correctly.**

Suggestion:
Doing one-sided label smoothing will really help preventing extreme extrapolation behavior in the discriminator .
Here is the resources https://arxiv.org/pdf/1701.00160.pdf

Rate this review
☆☆☆☆☆

✓ **The function model_opt is implemented correctly.**

Great job moving the creation of train_opt inside a with `tf.control_dependencies(tf.get_colletion(tf.GraphKeys.UPDATE_OPS))` statement. By doing so it will get the normalization layers created with `tf.layers.batch_normalization` to update the population statistics while training.

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

You need to multiply the batch image values by 2 `batch_images = batch_images * 2`. By doing so we normalize the batch_images to the range [-1, 1] as same as the returned values of the generator.

Suggestion:

Please generate the images at every 100 steps or so. Since GAN results are not correlated well with the loss functions, we want to see how the network is trained by looking at the generated images.

✓ **The parameters are set reasonable numbers.**

Good choices!

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

Though I can see faces, the generated images could be more clear. Please normalizing the batch_images in the train function and implementing other suggestions (optional), you will get better images.

RESUBMIT PROJECT

DOWNLOAD PROJECT

Best practices for your project resubmission

Ben shares 5 helpful tips to get you through revising and resubmitting your project.

Watch Video (3:01)

Rate this review