UDACITY

Logout

PROJECT

Generate Faces

A part of the Deep Learning Nanodegree Foundation Program

PROJECT REVIEW | CODE REVIEW | NOTES

Meets Specifications

SHARE YOUR ACCOMPLISHMENT

Great job!! Keep up the good work!!

Required Files and Tests

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

The required ipython notebook is submitted along with the helper files

✓ **All the unit tests in project have passed.**

All the unit tests are passed without any errors. Good job 👍

Build the Neural Network

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

The function "model_input" is coded correctly and it also meets the rank specification given to each of the placeholders viz real_input, z_input and learning_rate and is returned appropriately. Nice work here !! ✅

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

The function discriminator is implemented correctly. The kernel size used in all the tf.layers.conv2d function calls are coherent. Good that you have used kernel size as 5 which is ideal and maintained coherency across all the layers.👍
✅ Good use of dropout. I would suggest you have keep_probability as 0.8 that can help avoiding overfitting.
✅ Xavier's Initialisation of weights has further improved the network.

Tips:

💡 You can also use tf.contrib.layers.flatten() to flatten a layer. Follow this link:
https://www.tensorflow.org/api_docs/python/tf/contrib/layers/flatten

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

Great job here as well. 👏 The function generator is implemented correctly.

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

The function model_loss is implemented correctly. This function is coded correctly by utilizing the discriminator and generator functions defined above. The output tuple with discriminator loss and generator loss are returned perfectly. Great work!! 👍

Check out this paper(https://arxiv.org/abs/1606.03498) for more details on One-sided label smoothing.

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

The function model_opt is implemented correctly. The optimization operations are correctly done for GANs.The names are filtered properly and tf.trainable_variables is used to get all the trainable variables and returned after execution.
✅ Additionally, you have also made sure that all updates are computed before running optimizer by use of `control_dependencies`.

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**

✅ The model is built using model_inputs, model_loss, and model_opt.
✅ You are also re-scaling the batch_images which are of range -0.5 to 0.5 to -1.0 to 1.0.
✅ You have also taken care to maintain the range from -1 to +1while performing np.random.uniform(-1, 1, size=(batch_size, z_dim)) and did not use any random values.
✅ You have also run the optimization for the generator twice to make sure that the discriminator loss does not go to zero.

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

You have chosen the ideal set of hyper parameters for your model. Well played👍. Reducing the `learning_rate` to **0.0002** could help the images getting generated better.

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

Overall it's great work 👍. After running the project, I could be able to be able to see the images getting generated very well.

DOWNLOAD PROJECT

RETURN TO PATH

Rate this review
☆☆☆☆☆

Student FAQ