PROJECT

Generate Faces

A part of the Deep Learning Nanodegree Foundation Program

PROJECT REVIEW | CODE REVIEW | NOTES

Meets Specifications

SHARE YOUR ACCOMPLISHMENT

Great re-submission! Excellent job implementing the functions and your network 🎉 I've added some suggestions to improve your output further. Congratulations on passing the project and graduating from the Deep Learning Foundations Nanodegree!! 👍

Hope you enjoyed working on this project and learned a lot from the Nanodegree! Keep up the good work 😊

Required Files and Tests

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

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

Great work, all your code runs well! 😄

Build the Neural Network

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

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

Nice job creating the discriminator with reusable variables that returns the output and logits. I have some suggestions to improve your code further:

- Did you try increase the number of filters for your last convolutional layer? For example, to 512?
- Also, I would suggest to play around a bit more with the the kernel size. In general for the kernel size we can say smaller + deeper is better. Small kernel sizes will help capture fine details of the image, while the number of filters (the dimensionality of the output space) gives the ability to learn more different details.

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

If you make the suggested changes to the discriminator you should also adjust your generator. In addition, I would advise you to use a stride of 1 for your last layer(s), this can help to remove the chessboard effect further.

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

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

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

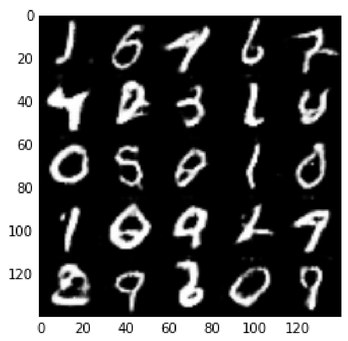
✓ **The parameters are set reasonable numbers.**

Great start! The hyperparameters have a big effect on the results of GANs. The generator and discriminator shouldn't overpower each other.

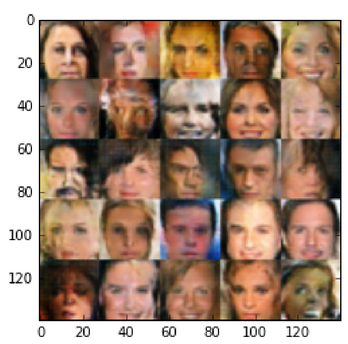
- The batch size is an important hyperparameter for GANs. A general rule in deep learning is that a bigger batch size is better (up until a certain point), however for GANs sometimes a smaller batch size works better. Therefore, 64 is perfect!
- You can experiment with slightly bigger learning rates, maybe around 0.0005.
- A beta1 of 0.5 is recommended here and you can leave the z_dim on 100.

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

Your output is already quite nice! I've played around with your code a bit (based on my suggestions above) and it resulted in the following output:



You may notice that the digits are less blurry. And the faces are also slightly more realistic:



I'm confident that by tweaking your networks a bit more you'll be able to create even more realistic images!

DOWNLOAD PROJECT

RETURN TO PATH

Student FAQ