Lewis Arnsten

In order to create my generator and discriminator I closely followed the architecture described in the pix2pix paper and attached sources (https://arxiv.org/abs/1611.07004). Specifically, I built my generator using a U-Net architecture where the input of size (256 x 256 x 3) is downsampled until a bottleneck layer (when input has shape 1x1x512), at which point the process is reversed. As outlined in the pix2pix paper, both my discriminator and generator layers are of the form convolution-BatchNorm-ReLu. For the encoder layers I used leaky ReLu activation, while for the decoder layers I used normal ReLu activation. Additionally, I included dropout in the first three decoder layers. Tanh activation was used for the last decoder layer. As the generator's goal is to minimize the difference between the generated image and a target image, I used L1 loss. Conversely, as the discriminator must make a binary classification as to whether an image is real or fake, I used binary cross entropy loss. This model achieves good results but takes a long time to run 4 epochs. The results are below.

Here is an image of 8 (edge, shoe, fake_shoe) triples and the FID achieved after 2 epochs:

```
Downloading: "https://github.com/mse
100%
22.128496163890986
```

Here is an image of 8 (edge, shoe, fake_shoe) triples and the FID achieved after 4 epochs:

```
import fid
eval_dataset = Edges2ShoesData
eval_dataloader = DataLoader(e
print(fid.get_fid(eval_dataloa
```

```
/usr/local/lib/python3.7/dist-
  cpuset_checked))
19.58677896594304
```