CONTENTS

Chapter 1

Training command and details

The command I used was mostly adapted from the README file in the sean directory. The paths to the images and labels were relative paths to the 2000 training images and labels I randomly (seeded) sampled from the CelebA-HQ dataset. As with SEAN these were cropped to 256x256 during training.

```
python3 train.py --name mike_crop_subset --load_size 256 --crop_size 256
↪  --dataset_mode custom --label_dir path/to/my/train/labels --image_dir
↪  path/to/my/train/images --label_nc 19 --no_instance --batchSize 2 --norm_mode
↪  clade
```

The difference with the SEAN command is that I enabled the CLADE $\text{norm}_{\text{mode}}$, which is specific to CLADE without it SPADE resblks are used.

```
# From CLADE/options/base_options.py
parser.add_argument('--norm_mode', type=str, default='spade', help='[spade | clade]')
```

More info on the training process is stored in text files in this directory.

- fid scores only contains two lines. Need to find out when the fid is being calculated. The options file mentions only that the fid is calculated every 10 epochs.
- iter text also only contains two lines. It is the 45 epochs times 2000 iterations the model has been trained for.
- loss log contains the loss function value per epoch.

Chapter 2

Adverserial loss term

The loss function is the same as with the pix2pixHD paper, instead they use a hinge loss form for the generator loss.

The general GAN loss function in pix2pixHD:

$$\min_G \max_D \mathcal{L}(G, D)$$

we use a multiscale discriminator by default, which you can check in the multiscale discriminator class in SPADE.

```python
# From models/networks/discriminator.py
class MultiscaleDiscriminator(BaseNetwork):
    @staticmethod
    def modify_commandline_options(parser, is_train):
        parser.add_argument('--netD_subarch', type=str, default='n_layer',
                            help='architecture of each discriminator')
        parser.add_argument('--num_D', type=int, default=2,
                            help='number of discriminators to be used in multiscale')
        opt, _ = parser.parse_known_args()

        # define properties of each discriminator of the multiscale discriminator
        subnetD = util.find_class_in_module(opt.netD_subarch + 'discriminator',
                                            'models.networks.discriminator')
        subnetD.modify_commandline_options(parser, is_train)

        return parser
# [...]
```

So the general loss form is actually (also in SEAN/SPADE),

$$\min_{E,G} \max_{D_1,D_2} \sum_{k=1,2} \mathcal{L}_{GAN}(E,G,D_k)$$

which is the minimax game objective. The objective function $\mathcal{L}$ is given by,

$$\mathcal{L}_{GAN}(G,D) = \mathbb{E}_{(s,x)}\left[\log D(\boldsymbol{s},\boldsymbol{x})\right] + \mathbb{E}_{s}\left[\log(1 - D(\boldsymbol{s}, G(\boldsymbol{s})))\right]$$

Where s is the label map, and x is the image.

Note that D is a (set of) fully convolutional network(s) with a sigmoidal activation function at the end (only in pix2pix paper, or original gan$_{\text{mode}}$ loss in SPADE project). This means that the range of D should be $[0,1]$. Where *one* means real and *zero* means fake.

2.1 Hinge version

Now in later papers (SPADE and its derivatives) a hinge form was used, without any sigmoid predictions. This is best explained in the SEAN paper,

$$\begin{aligned}\mathcal{L}_{GAN} = \mathbb{E}\left[\max(0, 1 - D_k(\boldsymbol{s},\boldsymbol{x}))\right] \qquad \{\} \\ + \mathbb{E}\left[\max(0, 1 + D_k(\boldsymbol{s},, G(\boldsymbol{s})))\right] \qquad \{\}\end{aligned}$$

Where again s is the label map and x is the real image. You can see that there are two hinge terms, the real and fake discriminator loss.

This is equivalent to the following (Zhang et al. 2019: SAGAN):

$$\begin{aligned}\mathcal{L}_D = -\mathbb{E}_{(s,x)}\left[\min(0, -1 + D(\boldsymbol{s},\boldsymbol{x}))\right] \qquad \{\} \\ - \mathbb{E}_{s}\left[\min(0, -1 - D(G(\boldsymbol{s}),\boldsymbol{s}))\right] \qquad \{\} \\ \mathcal{L}_G = -\mathbb{E}_{s}\left[D(G(\boldsymbol{s}),\boldsymbol{s})\right] \qquad \{\}\end{aligned}$$

Where $\mathcal{L}_G$ is the generator loss, this is important, because we are training stepwise the generator and discriminator. One step the $\mathcal{L}_D$ is computed and $\mathcal{L}_G$ in the other.

It can be shown that this equation converges to 2 , and that is equivalent to pushing the generated image to the separating hyperplane, and optimising the hyperplane margins for the discriminator (geometric gan paper).

The intuition for this is that when the probability distribution of the real images and fake images are equivalent, or the reverse KL-divergence $KL\left[p_g||q_{data}\right]$ is minimised [1]

2.2 SPADE implementation

The actual output of a forward pass through the MultiScaleDiscriminator is a nested list of dimension, $\mathrm{n_D}$ x $\mathrm{n_{LayersinD}}$. The discriminator is actually the two times application of a the normal Nlayer patchgan discriminator. It is a four layer fully convolutional network. The first time the input is normal, the second time the input is downsampled.

This nested list is used in the discriminate method of the pix2pix model class.

This tensor is fed to the $\mathrm{divide_{pred}}$ method of pix2pix to give the predictions to the loss class.

Chapter 2

Bibliography

[1] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral Normalization for Generative Adversarial Networks. *arXiv:1802.05957 [cs, stat]*, February 2018.