Style-Based GAN in PyTorch

Update (2019/09/01) I found bugs in the implementation thanks to @adambielski and @TropComplique! (https://github.com/rosinality/style-based-gan-pytorch/issues/33, https://github.com/rosinality/style-based-gan-pytorch/issues/34) I have fixed this and updated checkpoints

Update (2019/07/04)

- Now trainer uses pre-resized lmdb dataset for more stable data loading and training.
- Model architecture is now more closely matches with official implementation.

Implementation of A Style-Based Generator Architecture for Generative Adversarial Networks (https://arxiv.org/abs/1812.04948) in PyTorch

- Demo and Docker image on Replicate

Usage:

You should prepare lmdb dataset

```
python prepare_data.py –out LMDB_PATH –n_worker N_WORKER DATASET_PATH
```

This will convert images to jpeg and pre-resizes it. (For example, 8/16/32/64/128/256/512/1024) Then you can train StyleGAN.

for celebA

```
python train.py –mixing LMDB_PATH
```

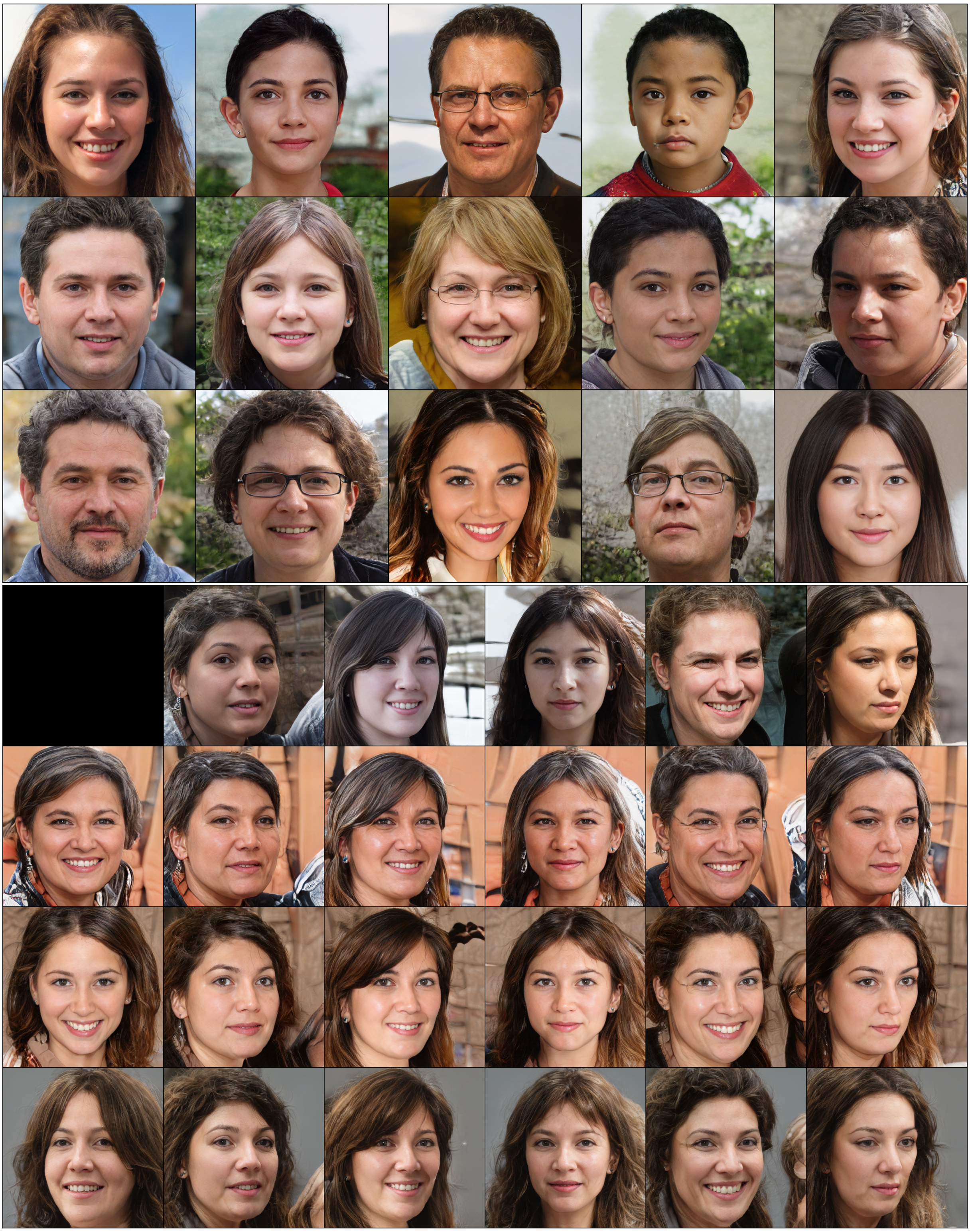
for FFHQ

```
python train.py –mixing –loss r1 –sched LMDB_PATH
```

Resolution	Model & Optimizer
256px	Link
512px	Link
1024px	Link

Model & Optimizer checkpoints saved at the end of phases of each resolution. (that is, 512px checkpoint saved at the end of 512px training.)

Sample



512px sample from the generator trained on FFHQ.

Old Checkpoints

Resolution	Model & Optimizer	Running average of generator
128px	Link	100k iter Link
256px	Link	140k iter Link
512px	Link	180k iter Link

Old version of checkpoints. As gradient penalty and discriminator activations are different, it is better to use new checkpoints to do some training. But you can use these checkpoints to make samples as generator architecture is not changed.

Running average of generator is saved at the specified iterations. So these two are saved at different iterations. (Yes, this is my mistake.)