IFT6135 Assignment4, Practical part

Xiao Fan (20086722)
Zhibin Lu (20091078)

April 20, 2018

Image Generation

2. Model

In this assignment, we chose to implement Generative Adversarial Networks (GANs).

3. Architecture

For the generator with transposed convolution, we used the architecture introduced in [1]. For nearest-neighbour upsampling and bilinear upsampling schema, we just replaced the transposed convolution layer with a corresponding upsampling layer and a regular convolution layer. For the discriminator, we used a symmetric structure.

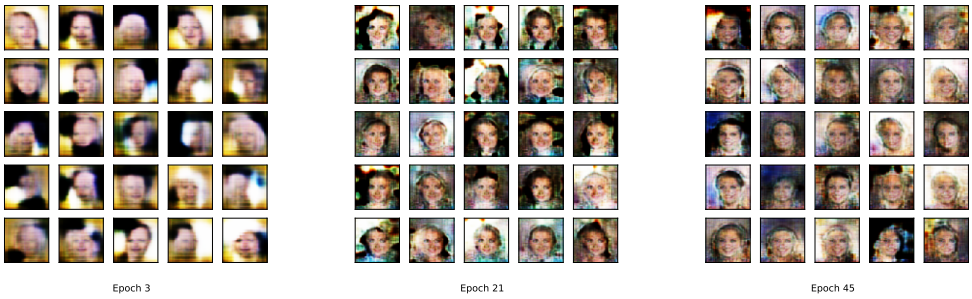
We only used 10k examples in the Dataset CelebA [2] to train our models because of limit of time and GPU resource. We used mini-batch size of 128, learning rate of 0.0002, latent dimension of 100 and Adam optimizer with coefficient of (0.5,0.999). Figure 1 shows the faces generated during the training with different schemes. With transposed convolution, images generated always have a strange checkerboard pattern of artifacts. In our experiment, since the image quality is poor, this phenomenon is not very obvious. But compared to nearest-neighbour upsampling and bilinear upsampling, the deconvolution tends to generate images with stronger colour. We may consider it as en effect of overlapping in deconvolution.

Images at epoch 3 may give us some intuitions about the difference of these three schemes. Bilinear upsampling generates smoother images. Also epoch 45 of bilinear seems worse than epoch 21, we may have problem of overfitting.

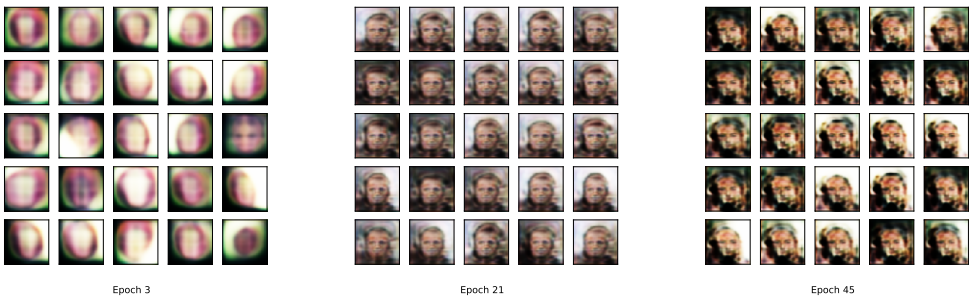
In all the three methods, we observed mode collapse, especially with bilinear-upsampling which gives almost one mode in the generated images. Also, we did not observe image quality improvement with upsampling methods as expected before experiment. In practice, models with upsampling methods are less stable and harder to train.

This fact can be illustrated with figure 2. Generator loss in nearest-neighbour upsampling and bilinear upsampling has a higher value and hard to decrease with more epochs. It is always above discriminator loss which means generator in these models has less capacity and can hardly compete with the discriminator. As a result, images generated has poorer quality and more severe mode collapse problem.

(a) Deconvolution (transposed convolution) with paddings and strides.

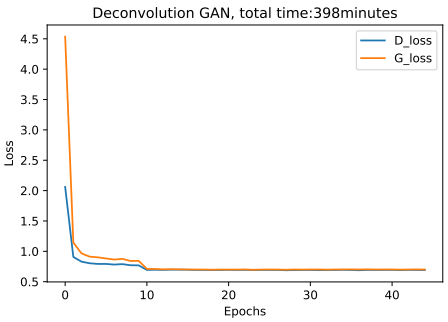


(b) Nearest-Neighbour Upsampling followed by regular convolution.

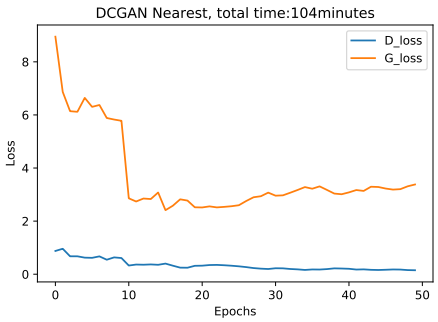


(c) Bilinear Upsampling followed by regular convolution

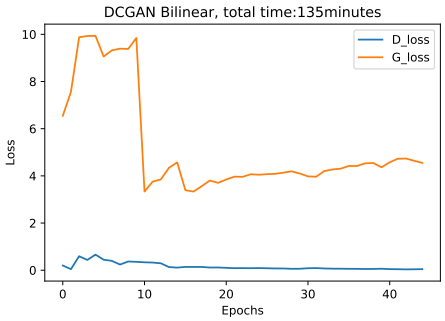
Figure 1: Images generated by different schemes to increase feature map size.



(a) Deconvolution (transposed convolution) with paddings and strides.



(b) Nearest-Neighbour Upsampling followed by regular convolution.

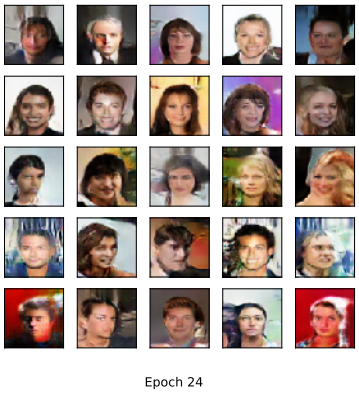


(c) Bilinear Upsampling followed by regular convolution

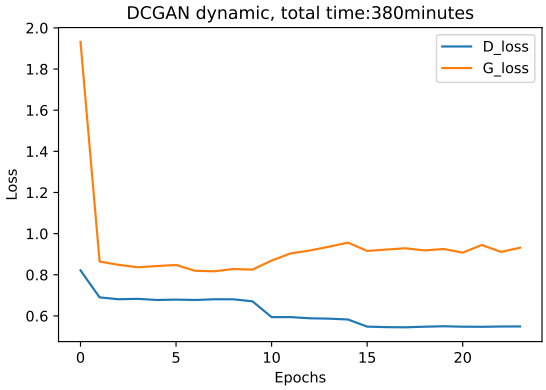
Figure 2: Generator loss and discriminator loss against training time.

Improve DCGAN with more sophisticated training process We made some efforts to improve images quality of DCGAN by different training process, for example several discriminator updates before one generator update or dynamically adjust critic number. One method exceptionally gives much better result than vanilla training procedure.

We adjust the critic number dynamically during the training. Critic number here is the number of discriminator/generator updates during one epoch. For each mini-batch, when training the discriminator, we want the D loss to be smaller than 0.69, because the optimal value of discriminator is $D^*(x) = \frac{p_r x}{p_r(x)+p_g(x)} = 0.5$ when fake data is perfect and $log(0.5) = -0.693$; similarly, during generator update, we train the generator until G loss is close to 0.5 which means the generator is nearly perfect. This procedure keeps an equilibrium between discriminator and generator and avoids neither of them to be too strong or too weak and can improve together as much as possible. Experiment results as in figure 3 show that this method is able to generate better images. However, the problem is that it takes much more time to train.



(a) Images generated with dynamic training algorithm



(b) Training loss

Figure 3: Improved image quality with dynamic training algorithm.

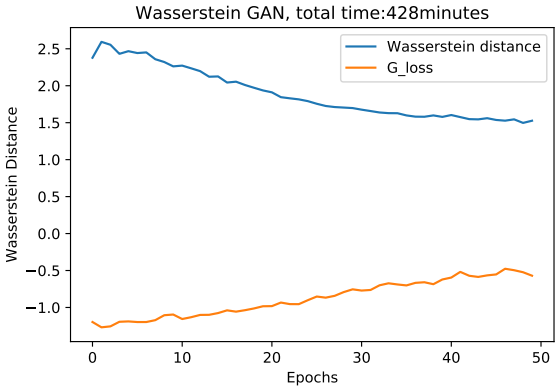
4. Variants

We implemented WGAN introduced in this paper [3]. As suggested in the paper, we clipped the weights between -0.01 and 0.01. Transposed convolution is used in following questions.

Compared with DCGAN, WGAN is easier to train and doesn't need careful balance between discriminator and generator as in DCGAN. The DCGAN is very unstable during training. It is often the case that the generator quickly becomes too powerful, causing the discriminator fail to progress, or vice versa. We need to carefully choose the hyper-parameters, e.g. critic numbers, learning rate. While in WGAN, the training is faster and easier because the new objective function - Wasserstein distance is more appropriate and introduces benefits like stability.

Figure 4 shows the images samples generated by WGAN at epoch 50 and training procedure. With more epochs, the Wasserstein distance is decreasing as expected.

(a) Images generated by WGAN



(b) Wasserstein distance against training

Figure 4: Images generated and loss of WGAN.

5. Qualitative Evaluations

(a) Visual samples

Figure 5 are visual samples of vanilla model (DCGAN) and variant model (WGAN).

(a) Images generated by DCGAN

(b) Images generated by WGAN

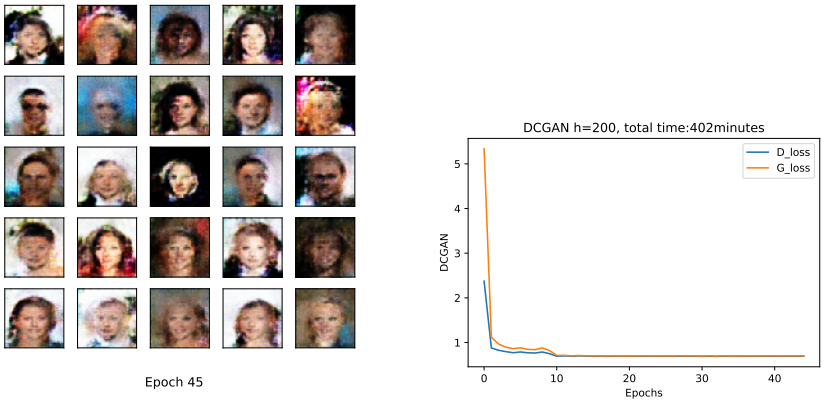
Figure 5: Visual samples of DCGAN and WGAN.

As expected, WGAN has better performance in mode collapse problem. Images generated are quite diverse. While in DCGAN, there are several images that are similar. With respect to blurriness, images generated by WGAN are clearer with less blurriness. But some faces still have distortions and defective features. Wasserstein distance solves the problem of vanishing gradient in JS divergence. It stabilize the training process and makes the model easier to train. Generator and discriminator

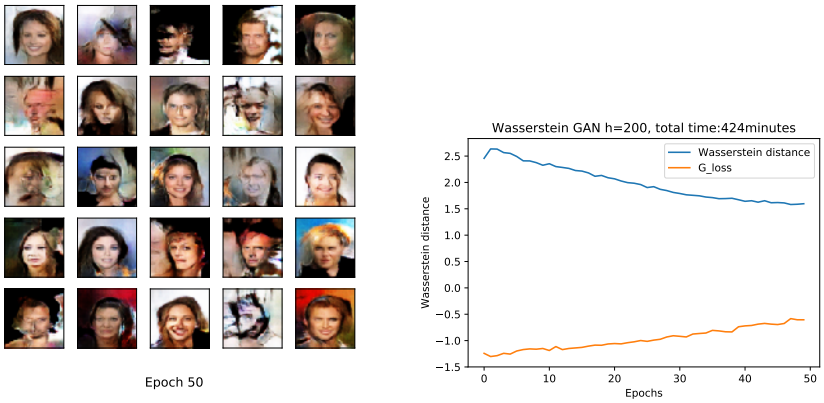
are well balanced, so there is less probability to collapse. But image quality also relates to model capacity, inference algorithm or training time. WGAN is not able to improve in these aspects. Hierarchical models e.g. auto-regressive model may be used to improve model capacity and gets better inference.

(b) Different dimension of latent space

We also tried to train DCGAN and WGAN with latent dimension of 200. Figure 6 shows the image samples and loss with latent dimension of 200. We didn't observe much improvement with increased dimension of latent space, contrarily, some images get worse. We may consider the dimension of latent space as the number of features needed to create an image. Too many latent variables may lead to problem of overfitting that one latent variable actually represents some specific examples without learning global features. However, models with too few latent variables don't have enough capacity. In our experiment, latent dimension of 100 suffices and we observed good images generated by the network. Increased latent dimension don't get better performance since the model capacity has saturated and it may lead to overfitting.



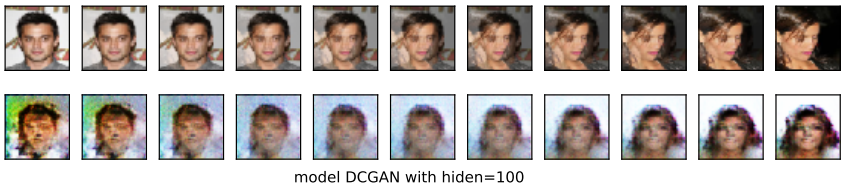
(a) DCGAN



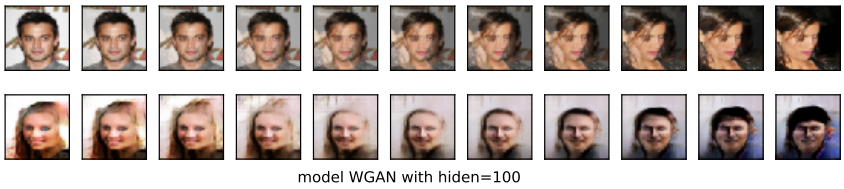
(b) WGAN

Figure 6: DCGAN and WGAN results with latent dimension of 200.

(c) Interpolate between images



(a) DCGAN



(b) WGAN

Figure 7: First line is interpolation with $x' = \alpha x_0 + (1-\alpha)x_1$, second line is $z' = \alpha z_0 + (1-\alpha)z_1$.

Figure 7 visualizes the interpolation effect of two schemes. Effect of $x' = \alpha x_0 + (1-\alpha)x_1$ is the same to superpose two images. As α close to 0.5, we can clearly see the existence of two overlapped images and it makes the new image really weird. The interpolation of latent variables demonstrates a better and realistic merged effect. For example, the direction of z_0's face slowly turns to the direction of z_1's face; z_0's hair colour gradually changes to z_1's hair colour. WGAN generates more natural images after interpolation compared to DCGAN.

6. Quantitative Evaluations (GAN)

We implemented Inception score and Wasserstein distance in this part.

(a) 1.98±0.05 (b) 2.86±0.07 (c) 2.37±0.11 (d) 2.67±0.12

Figure 8: Image quality and inception score (mean ± std).

We used pretrained inception_v3 model in torchvision as the inception model. As seen in figure 8, inception score correlates well with human judgment of image

Models	Inception score
DCGAN+transposed convolution Epoch21	2.701 ± 0.081
DCGAN+Bilinear Epoch33	1.870 ± 0.014
DCGAN+transposed convolution (hidden=200) Epoch45	3.241 ± 0.033
WGAN Epoch45	2.966 ± 0.060
WGAN (hidden=200) Epoch45	2.850 ± 0.054

Table 1: Inception score of different models

quality. The first image is very blur and has the lowest score, while the fourth is clearer and has a higher score. However, inception score is always reliable. For example, the second image has poor quality but has a high score. Table 1 lists the inception score of different models.

The inception score makes an assumption that good quality images can be correctly classified but has no measure on image diversity. So generator with severe mode collapse can still have high inception score. The pretrained inception model is trained on ImageNet, so the result may be misleading for usage beyond ImageNet like in our case. Moreover, the hyper-parameters like the split number, the number of examples can influence the value of inception score. As a conclusion, inception score gives us a global idea of image quality but is not a perfect metric and can be misleading sometimes.

Concerning Wasserstein distance, as in figure 4, the decreasing distance demonstrates the improving quality of generated images. We listed the Wasserstein distance of some models we trained in previous questions in table 2.

Models	Wasserstein distance
DCGAN+transposed convolution Epoch21	0.265
DCGAN+Bilinear Epoch33	0.564
DCGAN+transposed convolution (hidden=200) Epoch45	0.331
WGAN Epoch45	0.257
WGAN (hidden=200) Epoch45	0.266

Table 2: Wasserstein distance of different models.

Wasserstein distance gives better evaluation on model performance. Models with latent dimension of 200 has larger distance which is coherent with our observation that images become worse. DCGAN with bilinear upsampling has bad image quality and has a large distance. WGAN has the smallest distance and it actually produces the best images. Wasserstein distance evaluates the distance between real data distribution and generated data distribution, so is a better metric theoretically. However, we only get estimated value of this metric.

References

[1] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *CoRR*, vol. abs/1511.06434, 2015. [Online]. Available: http://arxiv.org/abs/1511.06434

[2] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of International Conference on Computer Vision (ICCV)*, 2015.

[3] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, D. Precup and Y. W. Teh, Eds., vol. 70. International Convention Centre, Sydney, Australia: PMLR, 06–11 Aug 2017, pp. 214–223. [Online]. Available: http://proceedings.mlr.press/v70/arjovsky17a.html