University of Toronto

DCGAN and CycleGAN analysis, emoji transfer

YiFei Gu

1. **Deep Convolutional GAN (DCGAN)**

   1) **Implement the Discriminator of the DCGAN:**

      i. Padding: use the formula $n_{out} = \lfloor \frac{n_{in}+2p-k}{s} \rfloor + 1$, with $k = 5, s = 2$, and $2n_{out} = n_{input}$ as suggested in the handout, we get a padding size of 2. Since n_{out} =1 in the last layer, we get a padding size of 1 for the last layer of `DCDiscriminator`.

   2) **Experiment:**

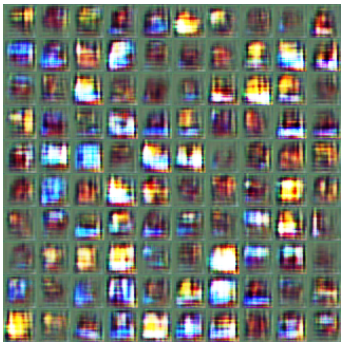

Figure 1: with 200 iterations

Figure 2: with 2600 iterations

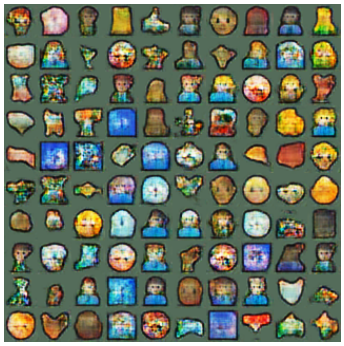

Figure 3: with 5000 iterations

- As shown above, figure 1 represents samples from early in training, figure 2 is the samples during middle stage of the training, figure 3 includes samples from later in training, with 5000 iterations, which gives relatively decent results.
- As noticed, with more iterations of training, the samples improves with better color accuracy, shape accuracy and resolution-wise, samples from later training provides better resolution, relative to early training, which provides readability of those emojis.
- note that if `batch_norm = False` is applied to the final layer of `DCGenerator`, the output of the samples become really dim, which is undesired but is left as an option. In the model above, `batch_norm = False` is only applied to the final layer of `DCDiscriminator`.

2. **CycleGAN Experiments:**

   1) Below figures show samples from both iteration 200 and iteration 5000, for both generators:

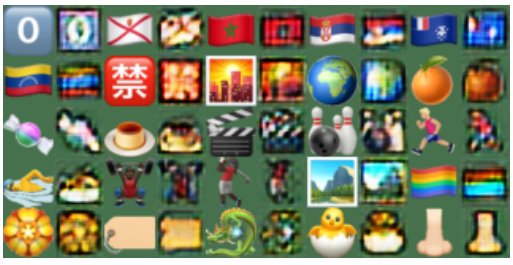

Figure 4: X-Y 200 iterations

Figure 5: X-Y 5000 iterations

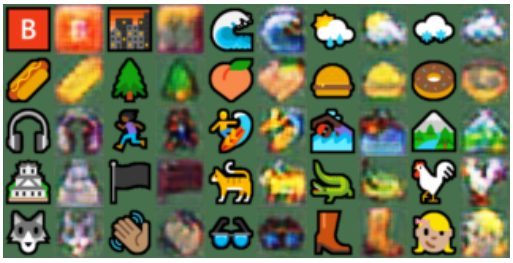

Figure 6: Y-X 200 iterations

Figure 7: Y-X 5000 iterations

As shown above, the colour and shape accuracy of samples quality improve as number training iterations increases, provided with better resolution and readability.

2) Some results with changed random seed with `SEED = 1`, comparing to `SEED = 4`:

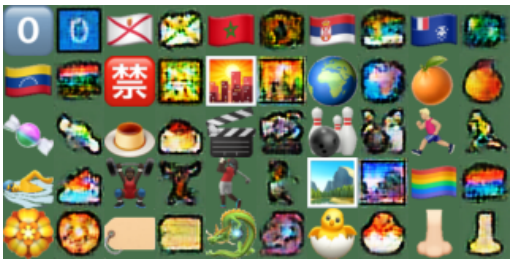

Figure 8: X-Y 5000 iterations with seed = 1

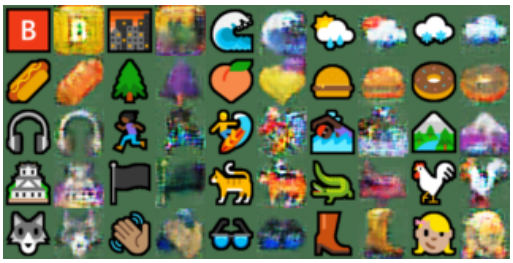

Figure 9: Y-X 5000 iterations with seed = 1

- Visually, there exists some colour difference between different seeds for 5000 iterations. As shown above, one can notice that red in original sample tends to be yellow in the generated samples, in addition, yellow tends to be red, and green tends to be purple in the generated samples.
- The reason why there exists a difference is that for different seed, weight initialization is different, also for the shuffling of images, with different initial weight, colour will be different.

3) Results without the cycle-consistency loss, by setting `lambda_cycle = 0` : Below figures show samples from both iteration 200 and iteration 5000, for both generators:

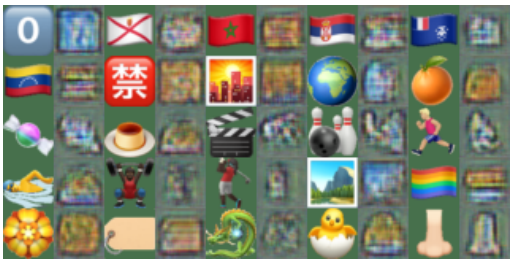

Figure 10: X-Y 200 iterations with $\lambda_{cycle} = 0$

Figure 11: X-Y 5000 iterations with $\lambda_{cycle} = 0$

Figure 12: Y-X 200 iterations with $\lambda_{cycle} = 0$

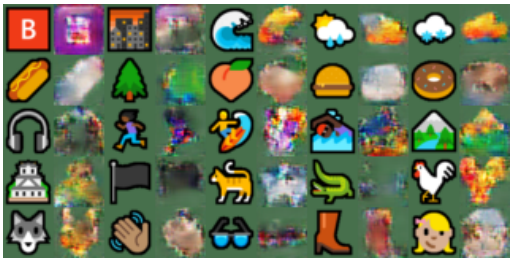

Figure 13: Y-X 5000 iterations with $\lambda_{cycle} = 0$

As shown above, without the cycle-consistency loss, the network performed poorly, image seemed blurry and without colour accuracy, even after 5000 iterations. Note that without cycle-consistency loss, the network is not aware of the reconstruction loss accounted in the generator loss, thus produced undesired results.

Doubling the original λ_{cycle} with `lambda_cycle = 0.03` gives:

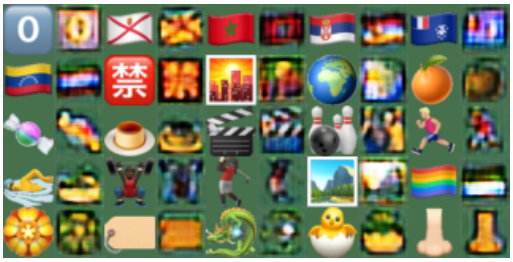

Figure 14: X-Y 200 iterations with $\lambda_{cycle} = 0.03$

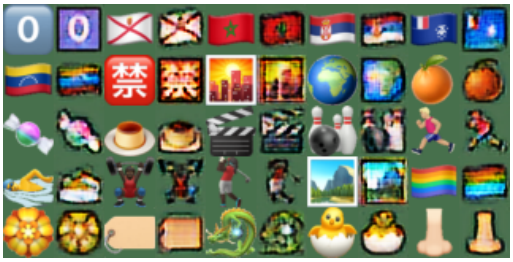

Figure 15: X-Y 5000 iterations with $\lambda_{cycle} = 0.03$

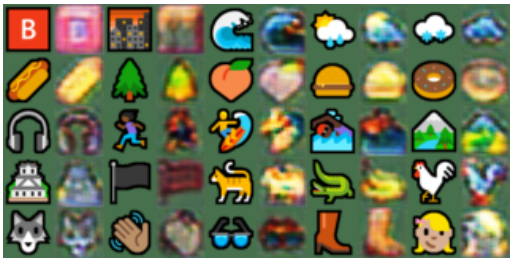

Figure 16: Y-X 200 iterations with $\lambda_{cycle} = 0.03$

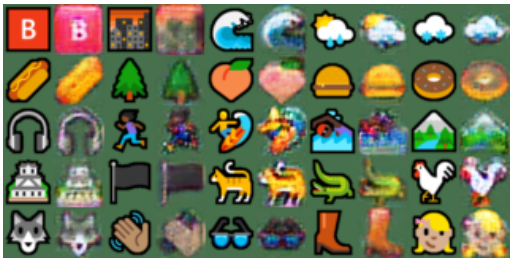

Figure 17: Y-X 5000 iterations with $\lambda_{cycle} = 0.03$

As shown above, with a $\lambda_{cycle} = 0.03$, the generated samples are overall similar to which with a $\lambda_{cycle} = 0.015$ in terms of shape, however, the colour accuracy of the previous does not seemed to outperform the default settings, there are some deviation in colour accuracy with $\lambda_{cycle} = 0.03$. Since cycle-consistency loss is generated from reconstructed image and original image, scaling of λ_{cycle} in fact does not penalize the deviation of generated image, rather increase the total generated loss, in that way results in a colour inaccuracy.