Multiple-CycleGAN and its Application in Style Transformation of Chinese Character Fonts and City Maps

Zheng Chuanyang(17251311), Gong Yikai(17251567)

Contribution: Zheng Chuanyang: 50%; Gong Yikai: 50%

1. **Abstract**

We invent a technique called Multiple-CycleGAN, which increase the number of cycles to get more accurate result. This paper aims at comparing Multiple-CycleGAN and original CycleGAN. There are two datasets for this testing, LCCD (Large Chinese Character Dataset) and City Maps dataset. LCCD is our hand-made dataset containing 20000 pairs of Chinese characters with one Kaiti and one Xingkai in each pair (Xingkai and Kaiti are both fonts). City Maps is a common dataset, which contains 1000 pairs of city vertical photos and city maps. We adopted MSE loss and human-eye judgement to evaluate LCCD. It turns out that Multiple-CycleGAN significantly outperforms CycleGAN ($p<0.01$) in MSE loss, and Multiple-CycleGAN outperforms CycleGAN in 40 randomly chosen characters (21:19). MSE loss was adopted in City Maps testing, where CycleGAN significantly outperforms Multiple-CycleGAN ($p<0.01$) in City Maps dataset. In conclusion, Multiple-CycleGAN performs better than CycleGAN in LCCD but worse in City Maps dataset.

Key word: Multiple-CycleGAN; style transformation; Chinese character; City Maps

2. **Introduction**

2.1 Model:

CycleGAN is a famous generative adversarial neural network [1]. However, sometimes the difference between element A and A’ (after cycling A->B->A’) is not large enough to optimize according to the gradient. In Multiple-CycleGAN, we try to repeat the step of cycling (A->B->A’->B’->A”->…) to get a larger offset so that it is easier to optimize the model. In this paper, we choose to cycle twice in our model.

2.2 Dataset:

There are two datasets for testing, LCCD (Large Chinese Character Dataset) and City Maps dataset.

The majority of font transformation is on English database, but few on Chinese because the number of Chinese characters is thousands of times of English letters. In this project, we create a Chinese character dataset named LCCD (Large Chinese Character Dataset). It has two types of Chinese characters. One is hand-writing type (Xingkai), another is called Regular Script (Kaiti). Each type has 18901 training images and 2000 testing images, 41802 images in total. And each image in one type has its corresponding images in another type. In other words, the dataset has pair images. The test with LCCD is a challenge for both Multiple-CycleGAN and original CycleGAN. Nowadays, most researchers on Chinese character generation apply classic machine learning, which performance is not so perfect. We also notice that because there are only 26 letters in English and their features are obvious, an offset of some strokes will not affect what the letter is, but those offsets will affect a Chinese character seriously especially the structure, so that besides pay attention to the style of the characters, we also have to focus more on the correctness. To reach this goal, human-eye judgement should be applied as a complement. We conducted a questionnaire survey and got 148 feedbacks. Each survey contains 40 characters, 20 for pattern similarity and 20 for style similarity.

In order to test whether Multiple-CycleGAN can be applied to a wide range of goals, we also used a common dataset City Maps to test our model. City Maps contains 1000 pairs of city vertical photos and city maps. City vertical photos are real black-and-white photos, while city maps are consisted of lines and color lumps representing different objects such as streets, coast and green belts. It can be regarded as a complement of last experiment because it will generate colors besides black and white and the figures covers almost the whole picture (LCCD has lots of blank).

2.3 Summarize of the novelty

2.3.1 An improved model: Multiple-CycleGAN

2.3.2 A hand-made large dataset: LCCD

2.3.3 Evaluation method: Human-eye judgement included

3. **Methodology**

3.1 CycleGAN is a famous generative adversarial neural network. CycleGAN has two domains, which are domain A and domain B. And it has different generators and discriminators for different domains. More importantly, compared with original GAN, CycleGAN has cycle-consistency loss and identity loss. The additional loss obviously improves the performance of CycleGAN.

3.2 Model.

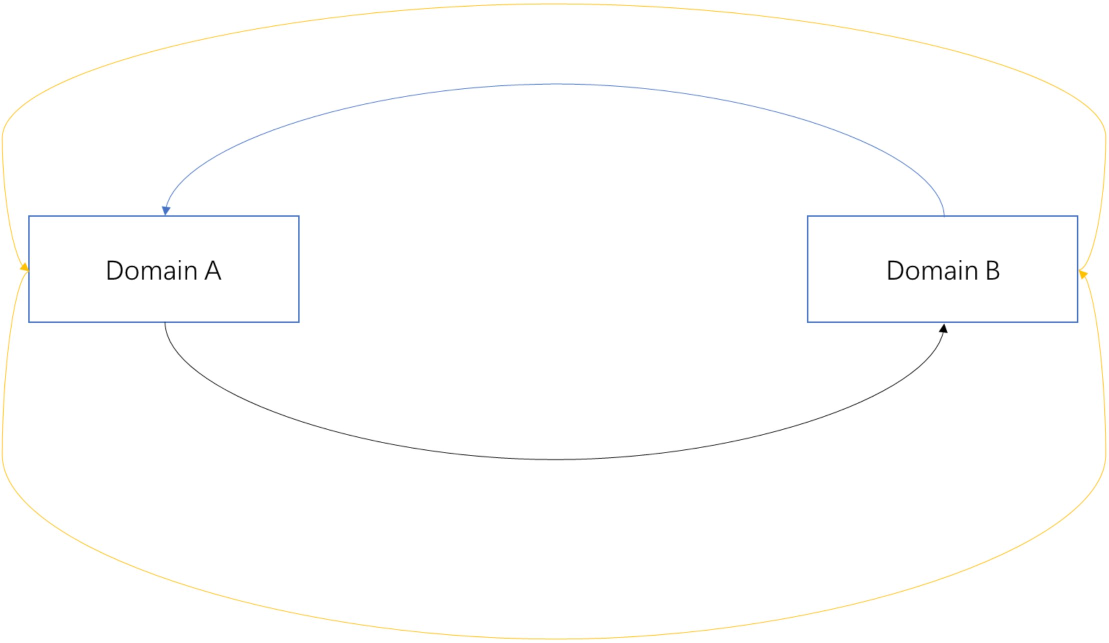


Figure 1 Black line points to first cycle. Yellow line points to the second cycle. The original CycleGAN only has the black line cycle. However, Theoretically, our model can have any number of cycles.

$$G_A: \text{Domain A} \rightarrow \text{Domain B}$$

$$G_B: \text{Domain B} \rightarrow \text{Domain A}$$

G_A is the generator which transfers an image in domain A to domain B. G_B is the generator which transfers an image in domain B to domain A. Let $\text{real}_A \in A, \text{real}_B \in$

B. Discriminator A $\mathrm{D_A}$ try to distinguish $\mathrm{real_A}$ and $\mathrm{fake_A}$, and discriminator B D_B try to distinguish $\mathrm{real_B}$ and $\mathrm{fake_B}$

For original CycleGAN, generated images are the following.

$$\mathrm{fake_A} = G_B(real_B)$$

$$\mathrm{fake_B} = G_A(real_A)$$

$$\mathrm{cycle_B} = G_A(fake_A)$$

$$\mathrm{cycle_A} = G_B(fake_B)$$

$$\mathrm{identity_A} = G_B(real_A)$$

$$\mathrm{identity_B} = G_A(real_B)$$

The corresponding loss function is the following:

$$loss_{D_B} = -(\log(\mathrm{D_B}(real_B) + \log(1 - \mathrm{D_B}(fake_B))$$

$$\mathrm{loss_{cycle_A}} = |real_A - cycle_A|$$

$$\mathrm{loss_{identity_A}} = |real_A - identity_A|$$

$$\mathrm{loss_{G_A}} = -\log(D_B(fake_B))$$

The $loss_{D_A}, loss_{cycle_B}, loss_{identity_A}$ and $loss_{G_A}$ is $similar.$

For our Infinite-CycleGAN, the generated images are the following. We regard $\mathrm{real_A}$ and $real_B$ as $\mathrm{cycle_{A_0}}$ and $cycle_{B_0}$ $seperately$

$$\mathrm{fake_{A_1}} = G_B(real_B) = G_B(cycle_{B_0})$$

$$fake_{B_1} = \mathrm{G_A(real_A)} = \mathrm{G_A(cycle_{A_0})}$$

$$cycle_{A_1} = G_B(fake_{B_1})$$

$$cycle_{B_1} = G\_A(fake_{A_1})$$

$$identity_{A_1} = G_{B(real_A)}$$

$$identity_{B_1} = G_A(real_B)$$

Then, we can continue cycling.

$$\mathrm{fake}_{\mathrm{A}_1} = G_B(cycle_{B_1})$$

$$\mathrm{fake}_{\mathrm{B}_1} = \mathrm{G}_{\mathrm{A}}(\mathrm{cycle}_{\mathrm{A}_1})$$

$$cycle_{A_2} = G_B(fake_{B_1})$$

$$cycle_{B_2} = G\_A(fake_{A_1})$$

$$identity_{A_2} = G_B(cycle_{A_2})$$

$$identity_{B_2} = G_A(cycle_{B_2})$$

The corresponding loss function is the following:

$$loss_{D_B} = -(\log(\mathrm{D}_{\mathrm{B}}(rea_B) + \log(D_B(cycle_{B_1}) + \log(1 - \mathrm{D}_{\mathrm{B}}(fake_{B_1}) + \log(1 - D_B(fake_{B_2}))$$

$$\mathrm{loss}_{\mathrm{cycle}_{\mathrm{A}}} = \left|real_A - cycle_{A_1}\right| + |cycle_{A_1} - cycle_{A_2}|$$

$$\mathrm{loss}_{\mathrm{identity}_{\mathrm{A}}} = \left|real_A - identity_{A_1}\right| + |cycle_{A_1} - identity_{A_2}$$

$$\mathrm{loss}_{\mathrm{G}_{\mathrm{A}}} = -(\log(D_B(fake_{B_1})) + \log\left(D_B(fake_{B_1})\right))$$

The $loss_{D_A}, loss_{cycle_B}, loss_{identity_A}$ and $loss_{G_A}$ is $similar.$

3.3 Pros and cons

3.3.1 Pros: data augmentation and loss magnification.

As our model can cycle many times, we can get fake_1, fake_2, fake_3…… fake_n, which is obviously a procedure of data augmentation. For many project, such as visible-to-thermal person re-identification, it can generate fake_1, fake_2……fake_n to train a classifier, while CycleGAN only generate fake_1. Meanwhile, we can notice that fake_(i+1) is based on fake_i, so the loss will accumulate and be magnified, which is easier to optimize.

3.3.2 Cons: Larger Memory Requirement

Assume CycleGAN uses M GB memory. If the model cycle N times, the requirements of our model is M*N GB memory. This is not friendly for laboratory which does not have enough memory, though they can change the batch size and input size. To solve this problem, user may run the cycles one by one until the n^{th} cycle. That is, run the first cycle, calculate the first cycle loss and backpropagation. Then repeat the steps until finishing the n^{th} cycle.

4. **Experimental Study**

4.1 Experimental Procedure

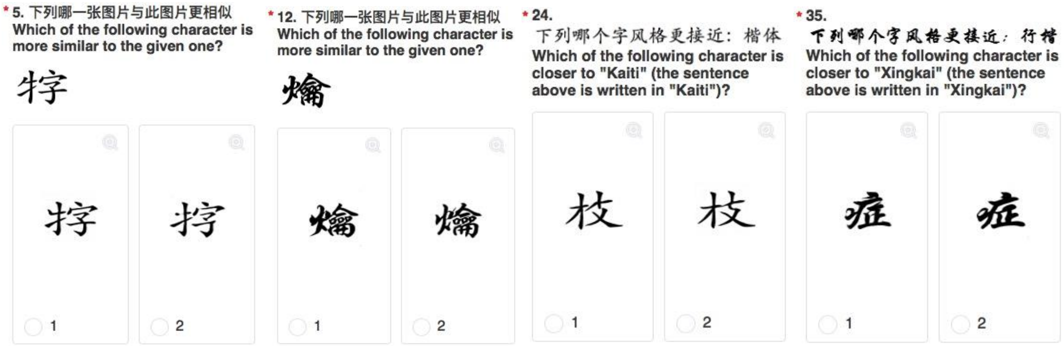
Just like original CycleGAN, for generative networks, our model has two stride-2 convolutions and 9 residual blocks. And for discriminative networks, our model use $70 X 70$ PatchGANs [2][3]**错误!未找到引用源。**. The aim is to classify the probability of $70 X 70$ images patches being real. For the discriminator architecture in patch-level, it needs less GPU memory and can work on any sized images in a fully convolutional fashion, compared to full-image discriminator.**错误!未找到引用源。**

For the training epoch, we determine to train 200 epochs, as cycleGAN training 200 epochs.

We conduct experiments on two datasets: LCCDB and City Maps. And we compare our approach against original CycleGAN method for unpaired image-to-image translation on the two datasets. And we will give corresponding Pytorch code.

4.2 Evaluation method

For LCCD testing, we use two ways to evaluate it. First, we use MSE loss to get the loss between fake images and ground truth. Second, we made a questionnaire and invite 148 volunteers to judge the quality of images generated by CycleGAN and Multiple-CycleGAN. In the questionnaire, there are two types of voting questions, 20 for pattern quality and 20 for style quality, 40 in total. In each type, there are 10 questions for Kaiti and 10 for Xingkai. These 40 characters in the questionnaire are randomly picked.



* 5. 下列哪一张图片与此图片更相似
Which of the following character is more similar to the given one?

○ 1 ○ 2

* 12. 下列哪一张图片与此图片更相似
Which of the following character is more similar to the given one?

○ 1 ○ 2

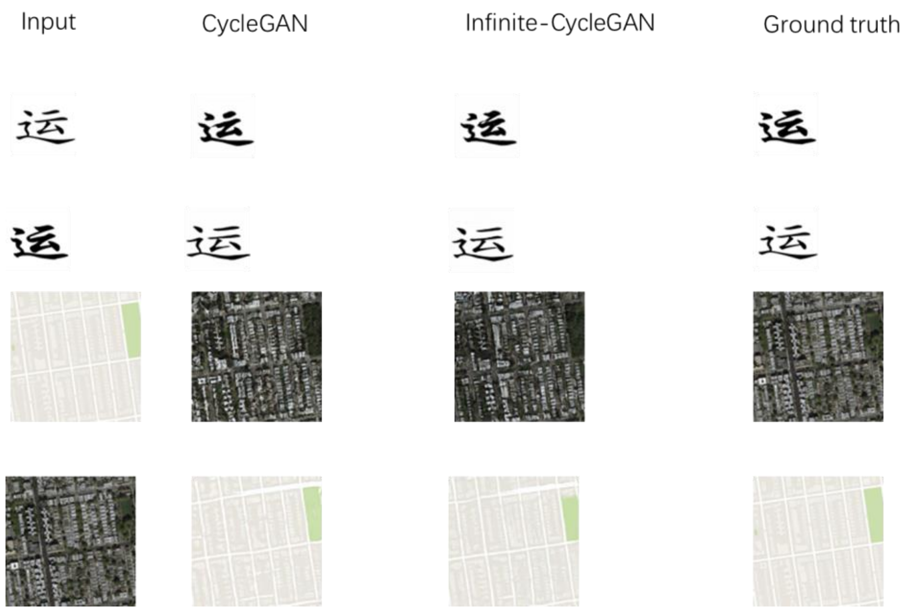
* 24.
下列哪个字风格更接近：楷体
Which of the following character is closer to "Kaiti" (the sentence above is written in "Kaiti")?

○ 1 ○ 2

* 35.
下列哪个字风格更接近：行楷
Which of the following character is closer to "Xingkai" (the sentence above is written in "Xingkai")?

○ 1 ○ 2

For Maps dataset, there exists two ways to evaluate: Amazon Mechanical Turk (AMT) and MSE loss. However, because of budget limit, we just simply use MSE loss to calculate the loss between generated images and the ground truth.



The corresponding MSE loss evaluation result of LCCD dataset is the following:

	Kaiti->XingKai	Xingkai->Kaiti
CycleGAN	0.3177	0.2979
Multiple-CycleGAN	0.3034	0.2981
Mean of d	0.0142	-0.0001
Standard deviation of d	0.0276	0.0190
t value of d	23.0975	-0.3436
p value of d	<0.01	>0.05

*(d=$\text{loss}_{\text{CycleGAN}}$-$\text{loss}_{\text{Multi-CycleGAN}}$)

And, we also prepare questionnaires for LCCD dataset. The result is the following.

Dataset	CycleGAN	Multiple-CycleGAN
LCCD	21	19

The correspond MSE loss evaluation result of **Maps** dataset is the following:

	Photo->Map	Map->Photo
CycleGAN	0.0229	0.1210
Multiple-CycleGAN	0.0244	0.1229

Mean of d	-0.0015	-0.0018
Standard deviation of d	0.0162	0.0260
t value of d	3.0378	2.3557
p value of d	<0.01	<0.05

*d=$\text{loss}_{\text{CycleGAN}}$-$\text{loss}_{\text{Multi-CycleGAN}}$

4.3 Experimental result

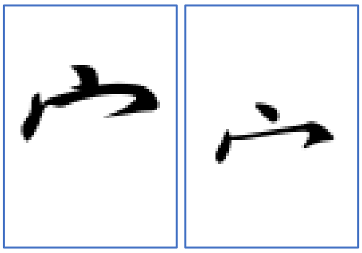
On LCCD dataset, using MSE loss, Multiple-CycleGAN outperforms CycleGAN because the loss is lower in Kaiti->Xingkai part(p<0.01) and the Xingkai->Kaiti part loss has no significant difference(p>0.05). And the human-eye judgement also proved our result. On Maps dataset, CycleGAN is better than Multiple-CycleGAN, as their MSE loss is lower at both Photo->Maps(p<0.01) and Maps->Photo(p<0.05).

In summary, Multiple-CycleGAN outperforms CycleGAN on LCCD and CycleGAN performs better on City Maps dataset. Here, we have three assumptions.

4.4 Discussion

4.4.1 About evaluation method

MSE loss was adopted in this evaluation. However, due to budget limit, Amazon Mechanical Turk (AMT) has not been implemented, which was applied in the original paper [1]. MSE loss is calculated pixel by pixel, so a tiny offset may cause a big loss.



In LCCD, for some characters, there are a relatively big offset between Kaiti and Xingkai. The change of position is hard to learn and it is not our point. However, MSE loss emphasizes it. It may affect the result. And since Multiple-CycleGAN outperforms on this dataset, it gives a further direction that whether Multiple-CycleGAN is more sensitive to position features.

For questionnaires, 148 volunteers are enough for this experiment, but all the volunteers are focused on 40 characters. The choice of the characters may affect the result. It is better to randomly pick different 40 characters to every volunteer.

It is obvious that blank in LCCD is much more than in City Map, it gives a further direction that whether Multiple-CycleGAN is more sensitive to structure features while CycleGAN is more sensitive to color lumps.

Due to time limit, the training is not well enough. On the one hand, repeat times may be not adequate. In this experiment, every element (character, photo, map) has been trained only one time. On the other hand, there are 200 epochs in LCCD in total but only 125 of them are able to run in one month. We do not have the confidence to say this model is coverage so there is still a doubt that whether we can definitely judge the models. In further study, the experiment will continue and be repeated and the result will be evaluated again.

Besides modifying the training and evaluation method, we will conduct experiments on cityscapes dataset and use corresponding evaluation python file to further evaluate our model. Cityscapes dataset does not need human-eye judgement and we already have corresponding evaluation file. A small problem is that it is not easy to download 6.6GB cityscapes dataset and successfully upload it to the machines.

5 Conclusion

In conclusion, Multiple-CycleGAN outperforms CycleGAN on LCCD (especially Kaiti->Xingkai) while CycleGAN performs better on City Maps dataset in both directions.

We will try to solve the problems above and conduct further experiment we have mentioned. Whether or not Multiple-CycleGAN is actually the better one, we just show what we have done in the past months. We are responsible for the authenticity of the experimental procedure, data and result.

Reference:

[1]. Zhu, J. Y., Park, T., Isola, P., & Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the IEEE

international conference on computer vision (pp. 2223-2232).

[2]. P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Imageto-image translation with conditional adversarial networks. In CVPR, 2017

[3]. C. Li and M. Wand. Precomputed real-time texture synthesis with markovian generative adversarial networks. ECCV, 2016