Calligraphy Style Transfer using Generative Adversarial Networks

Bharath Narasimhan
University of Massachusetts Amherst
bnarasimhan@cs.umass.edu

Abstract

In this project, we examine the problem of learning a mapping from one personalized handwriting style to another. We use two architectures - a) a conditional generative adversarial network (pix2pix) and b) a cycle-consistent generative adversarial network (CycleGAN) to generate handwritten English words of a particular style. We evaluate the effiicency of the proposed methods qualitatively as well as quantitatively.

1. Introduction

Style transfer of calligraphy is desirable for multiple reasons. In addition to aesthetics, it has applications to standardizing documents, e.g., doctor prescriptions and handwritten notes. Although handwriting generation is not as widely studied as handwriting recognition, previous approaches [6] in this area have been successful to an extent. They use long short-term memory recurrent neural networks for handwriting synthesis. Apart from a difference in objectives (style transfer versus style generation), they only focus on local representations of the characters rather than the overall style , and thus need to adjust the shapes, sizes, and positions of the strokes for every new character.

Figure 1. Objective : Learn a mapping G from the source style X to the target style Y given training samples $\{x_i\}_{i=1}^{N}$ where $x_i \in X$ and $\{y_j\}_{j=1}^{M}$ where $y_j \in Y$.

2. Related Work

2.1. Generative Adversarial Networks

GANs General adversarial networks [5] are powerful generative models which have achieved impressive results in many computer vision tasks, especially image-to-image translation [7]. GANs formulate generative modeling as a game between two competing networks: a generator network produces synthetic data given some input noise and a discriminator network distinguishes between the generator's output and true data. Formally, the game between the generator G and the discriminator D has the minimax objective:

$$\min_G \max_D \mathbb{E}_{x\sim\mathbb{P}_r}[\log D(x)] + \mathbb{E}_{z\sim\mathbb{P}_g}[\log(1 - D(G(z))], \tag{1}$$

where $x \sim \mathbb{P}_r$ are samples from the input data and $z \sim \mathbb{P}_g$ are the random noise samples, $G(z)$ are the generated images using the neural network generator G, and $D(\cdot)$ gives the probability of an input being real.

cGANs and pix2pix Unlike GANs which learn a mapping from a random noise vector to an output image, conditional GANs (cGANs) learn a mapping from a random noise vector to an output image conditioning on additional information. cGANs are capable of image-to-image translation since they can condition on an input image and generate a corresponding output image. Pix2pix [7] is a generic image-to-image translation algorithm using cGANs. It can produce reasonable results on a wide variety of problems. Given a training set which contains pairs of related images, pix2pix learns how to convert an image of one type into an image of another type, or vice versa.

CycleGANs Cycle-consistent GANs learn the image translation without paired examples [10]. Instead, it trains two generative models cycle-wise between the input and output images. In addition to the adversarial losses, cycle consistency loss is used to prevent the two generative models from contradicting each other. The default generator architecture of CycleGAN is ResNet, while the default discriminator architecture is a PatchGAN classifier.

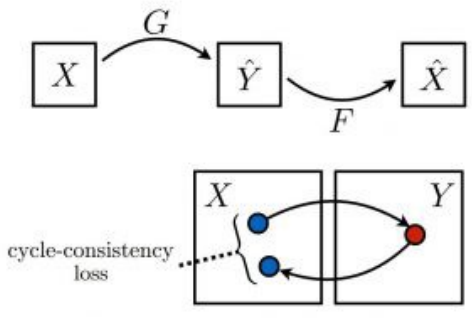

Figure 2. Cycle consistency loss

The CycleGAN objective is:

$$\begin{aligned}\mathcal{L}(G, F, D_X, D_Y) =& \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) \\ &+ \mathcal{L}_{\text{GAN}}(F, D_X, Y, X) \\ &+ \lambda \mathcal{L}_{\text{cyc}}(G, F) \end{aligned} \tag{2}$$

with

$$\begin{aligned}\mathcal{L}_{\text{cyc}}(G, F) =& \mathbb{E}_{x \sim p_{\text{data}}(x)}[F(G(x)) - x_1] \\ &+ \mathbb{E}_{y \sim p_{\text{data}}(y)}[G(F(y)) - y_1] \end{aligned} \tag{3}$$

where λ controls the relative importance of the two objectives. It aims to solve:

$$G^*, F^* = \arg \min_{G,F} \max_{D_x, D_Y} \mathcal{L}(G, F, D_X, D_Y) \tag{4}$$

2.2. Chinese character handwriting generation

Zi2zi Zi2zi [1] uses GAN to transform Chinese characters between fonts in an end-to-end fashion, assuming no stroke label or any other auxiliary information which is usually difficult to obtain. The network structure of zi2zi is based on pix2pix with the addition of category embedding for multiple fonts. This enables zi2zi to transform characters into several different fonts with one trained model. Zi2zi uses paired Chinese characters of the source font and the target font as the training data.
Another approach [1] studies the handwritten Chinese character generation problem with unpaired training data. It uses a DenseNet CycleGAN to learn a mapping from an existing printed font to a personalized handwritten style.

2.3. Image style transfer

Current image style transfer methods can be divided into two categories, namely descriptive neural methods based on image iteration and generative neural methods based on model iteration. Descriptive neural methods transfer the style by directly computing the gradient with respect to the source image and updating pixels in the image iteratively, while generative neural methods first optimize a generative model and produces the styled image through a single forward pass.

Neural style [4] is one of the most widely used descriptive neural methods for reproducing the content of an image with the style of another. It formulates style transfer as an optimization problem that combines texture synthesis with content reconstruction. Patch-based loss is added on top of content and style losses in [8, 3].

The drawback of descriptive neural methods is that the iterative updating algorithm only works for a single image, which makes it rather time-consuming if one would like to transfer the styles of many images. In contrast, generative neural methods are faster but usually generates poorer style transfer results.

It is difficult to define the content loss between characters in different styles since the strokes can be very different in positions and angles.

2.4. Generation with Recurrent Neural Networks

Long Short-term Memory recurrent neural networks [6] can be used to generate complex sequences with long-range structure, simply by predicting one data point at a time. The resulting system is able to generate realistic cursive handwriting in a wide variety of styles.

3. Approach

3.1. pix2pix

Given enough paired training examples, pix2pix should be able to learn the mapping from Style A to Style B, at least at the character level.
The pix2pix generator uses a UNet as shown in Figure 3 instead of an encoder-decoder architecture. It has *skip connections* directly connecting encoder layers to decoder layers which give the network the option of bypassing the encoding/decoding part if it does not have use for it. The pix2pix discriminator is of standard PatchGAN architecture.

3.2. CycleGAN

There are a few disadvantages to using pix2pix for the calligraphy style transfer problem :

- It is difficult to obtain a large set of paired training examples.
- It is important to learn overall style instead of mimicking every single character/word. This is because the

[1] https://github.com/kaonashi-tyc/zi2zi

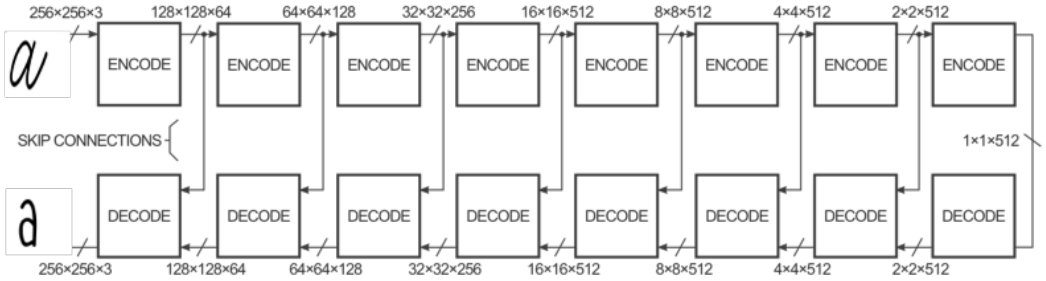

Figure 3. pix2pix generator architecture for paired image-to-image translation

user does not write the character/word in the same way always.

Cycle-consistent GANs (CycleGANs) learn image-to-image translation without paired examples and are applicable solutions to the problem at hand.
The CycleGAN generator consists of 9 ResNet blocks. The CycleGAN discriminator is of standard PatchGAN architecture.

4. Experiment

4.1. Datasets

IAM Handwriting Database The IAM Handwriting Database contains forms of handwritten English text which can be used to train and test handwritten text recognizers and to perform writer identification and verification experiments. The database contains forms of unconstrained handwritten text, which are scanned at a resolution of 300dpi and saved as PNG images with 256 gray levels. The database provides complete forms, text lines and extracted words.
There is no public dataset available for handwritten English characters. For experiments on handwritten characters, we collected 26 characters from two different styles *Aesthetik* and *Dorathy* manually from the Internet.

4.2. Network Setup

The PyTorch implementation for pix2pix and CycleGAN can be found here[2].
pix2pix Images were reshaped to 256x256, and paired together by horizontal concatenation. 26 character examples and 84 word examples were used for training after splitting into training, validation, and test sets. Training was done for 500 epochs, with a learning rate of $\alpha = 0.0002$ decaying to 0 after the 250^{th} epoch.

[2]https://github.com/eriklindernoren/PyTorch-GAN

CycleGAN Images were reshaped to 256x256. Data augmentation was performed using randomized flips.

4.3. Character pix2pix

The validation results are shown in Figure 5. We see that the pix2pix architecture is able to learn the character mapping quite well. This could be a possible solution to style transfer of handwritings that are not cursive in nature.

4.4. Word pix2pix

The word mapping is also learned satisfactorily, (although these are images from the validation set). With enough training data, it could be possible to use this for cursive styles. The progress of training over multiple epochs is shown in Figure 7

4.5. Character CycleGAN

Training was carried out with $\alpha = 0.0002, 0.002$. Training diverged in the latter case and the results for the former are shown in Figure 8. Training for a lareger number of epochs with hyperparameter tuning should produce better results theoretically.

4.6. Word CycleGAN

Training was carried out with $\alpha = 0.0002, 0.002$. Training diverged in both case and the results for the former are shown in Figure 9. Intermediate observations show minimal learning apart from font thickness.

5. Evaluation

5.1. Qualitative Analysis

We observe that the character pix2pix works well for almost all characters. Slight aberrations are seen in letters with a curve in them, like *'p'* and *'q'*. It was also observed that the dots above *'i'* and *'j'* were not generated at times

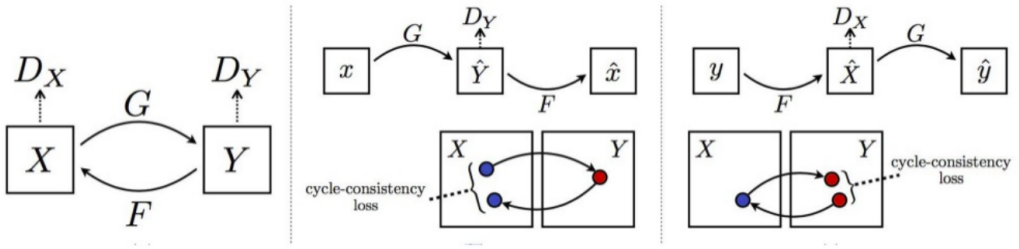

Figure 4. CycleGAN architecture for unpaired image-to-image translation

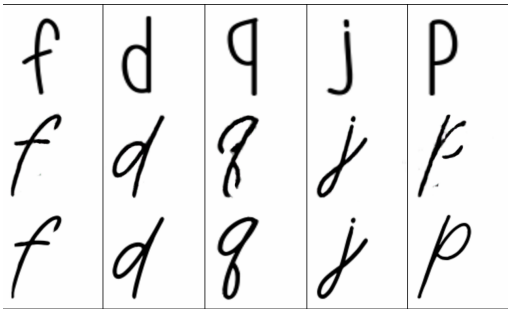

Figure 5. Generated character images from pix2pix - Rows 1 and 3 represent real images of Styles A and B respectively. Row 2 represents the generated validation image after training.

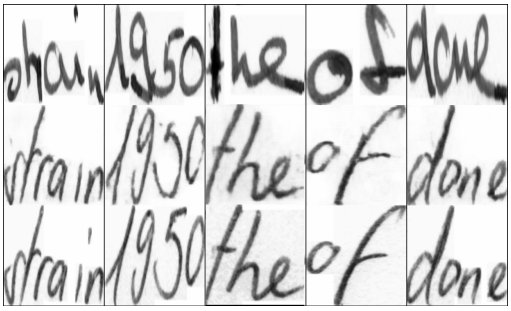

Figure 6. Generated word images from pix2pix - Rows 1 and 3 represent real images of Styles A and B respectively. Row 2 represents the generated validation image after training.

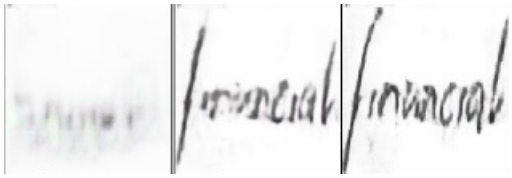

Figure 7. Validation images at epoch = $0, 250, 500$

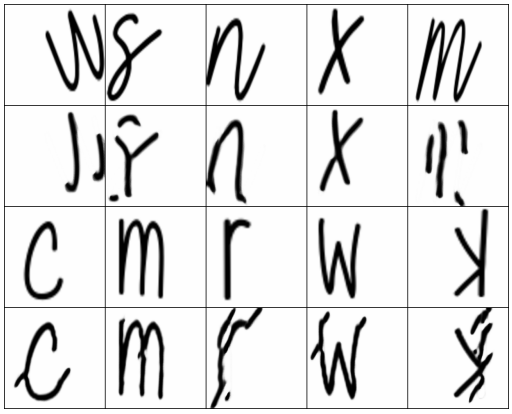

Figure 8. Generated character images from CycleGAN

for both character and word pix2pix.
The CycleGAN struggles to learn in both cases, perhaps because of the alignment of images and the large variance of the images. It should be possible to train a CycleGAN for characters, replicating the results for Chinese characters in [2]

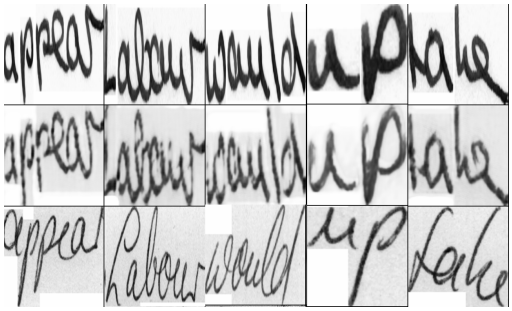

Figure 9. Generated word images from CycleGAN. The Rows 1 and 3 represent real images of Styles A and B respectively. Rows 2 and 4 represent the generated validation images after training.

5.2. Inception Score

The Inception Score is a metric for automatically evaluating the quality of image generative models [9] . This metric was shown to correlate well with human scoring of the realism of generated images from the CIFAR-10 dataset. The IS uses an Inception v3 Network pre-trained on ImageNet and calculates a statistic of the network's outputs when applied to generated images.

$$\mathrm{IS}(G) = \exp\left(\mathbf{x}\sim p_g \; D_{KL}(\, p(y|\mathbf{x}) \parallel p(y) \,) \right), \qquad (5)$$

where $\mathbf{x} \sim p_g$ indicates that $\mathbf{x}$ is an image sampled from p_g, $D_{KL}(p\|q)$ is the KL-divergence between the distributions p and q, $p(y|\mathbf{x})$ is the conditional class distribution, and $p(y) = \int_{\mathbf{x}} p(y|\mathbf{x}) p_g(\mathbf{x})$ is the marginal class distribution.

The authors who proposed the Inception Score aimed to codify two desirable qualities of a generative model into a metric:

1. The images generated should contain clear objects (i.e. the images are sharp rather than blurry), or $p(y|\mathbf{x})$ should be low entropy. In other words, the Inception Network should be highly confident there is a single object in the image.

2. The generative algorithm should output a high diversity of images from all the different classes in ImageNet, or $p(y)$ should be high entropy.

If both of these traits are satisfied by a generative model, then we expect a large KL-divergence between the distributions $p(y)$ and $p(y|x)$, resulting in a large Inception Score. From Table 1, we see that the inception score decreases from pix2pix to CycleGAN, and from characters to words respectively.

Model	IS
Character pix2pix	2.117
Word pix2pix	1.469
Character CycleGAN	1.210
Word CycleGAN	1.134

Table 1. Inception scores for trained models. Inception Score is a measure of how different the score distribution for a generated image is from the overall class balance

6. Conclusions

- In this project, we formulated the calligraphy style transfer problem as learning a mapping from one handwriting style to another.
- We used the pix2pix conditional GAN as well as the ResNet CycleGAN to generate characters as well as words.
- We find that the models with paired training examples (pix2pix) produced satisfactory results.
- It is however desirable to solve this problem efficiently with unpaired training examples, and this limitation needs to be addressed in the future.
- Preprocessing with proper alignment of images, keeping the aspect ratio same, could help train the CycleGAN better.
- Generation of more training data is desirable to improve the test accuracy of pix2pix as well as CycleGAN.

References

[1] B. Chang, Q. Zhang, S. Pan, and L. Meng. Generating handwritten chinese characters using cyclegan. *CoRR*, abs/1801.08624, 2018.

[2] J. Chang, Y. Gu, and Y. Zhang. Chinese typeface transformation with hierarchical adversarial network. *CoRR*, abs/1711.06448, 2017.

[3] T. Q. Chen and M. Schmidt. Fast patch-based style transfer of arbitrary style. *CoRR*, abs/1612.04337, 2016.

[4] L. A. Gatys, A. S. Ecker, and M. Bethge. A neural algorithm of artistic style. *CoRR*, abs/1508.06576, 2015.

[5] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.

[6] A. Graves. Generating sequences with recurrent neural networks. *CoRR*, abs/1308.0850, 2013.

[7] P. Isola, J. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *CoRR*, abs/1611.07004, 2016.

[8] C. Li and M. Wand. Combining markov random fields and convolutional neural networks for image synthesis. *CoRR*, abs/1601.04589, 2016.

[9] T. Salimans, I. J. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. *CoRR*, abs/1606.03498, 2016.

[10] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017.