Human to Anime face transfer using CycleGANs

Simon Döbele
sidobele@kth.se

Christos Frantzolas
frant@kth.se

Oliver Möller
olivermo@kth.se

Abstract

In this paper, we solve the task of neural style transfer (NST) using CycleGANs. First, we obtain a baseline CycleGAN, verifying that it works on the horses2zebras dataset. Second, we perform NST on an anime face dataset, in order to generate anime faces based on human faces. In order to find better generated images, several experiments were performed where the training set up (learning rate scheduling, optimizer, regularization, loss function) was altered. The resulting stylized images were evaluated by visual inspection, and thus the performance the model was judged on the basis of subjective features (e.g. artifacts, fidelity of the stylized images to the original). The CycleGAN model was shown to be able to perform this style transfer task, and not on a per-image basis, as previous NST models could. It is also worth noting that obtaining quality results required a significant amount of experiments and computation time.

1 Introduction

A common challenge in computer vision is image-to-image translation. Given a pair of images from the classes X and Y, the goal is to learn the mapping between them to transfer a new image from class Y to class X or vice versa. However, a typical problem in image-to-image translation is that paired data is often unavailable. Cycle Generative Adversarial Networks (CycleGAN) solve that problem and are capable of learning such mappings even from unpaired data.

To explore how CycleGANs work, we implemented our version and ran various experiments testing among others a different optimizer (Adam vs. SGD), regularization, linear vs. cyclic learning rate scheduling, as well as loss function weighting schemes. After obtaining sufficiently good results on the NST task, we employed a pre-trained image segmentation model, in order to apply the style transfer only on a select area of the images (in our case, human faces).

We found that the CycleGAN, when trained for a sufficient amount of epochs on a moderately sized dataset of unpaired images (approx. 1000-2000 images/domain) can perform Neural style transfer with relative success, even on a complex domain, such as human faces. This transformation between two domains is not always performed with the same level of success in both ways; one transformation is harder for the network to learn than the other. Also, to obtain results of sufficient quality, one must employ a loss weighting scheme for the generator networks loss which is suitable for a particular domain/dataset.

2 Related Work

2.1 Neural Style Transfer

In Image-to-Image translation, the goal is to learn a mapping between an input image from class X and an output image from class Y to translate a given image from one class to the other. An example task would be to transfer photographs to sketches or vice versa. A specific way to perform Image-to-Image translation is Neural Style Transfer [1]. Neural style transfer aims to generate new images by learning the content of one image and embedding it in the style of another image. A typical application is the transformation of paintings into photographs.

2.2 Generative Adversarial Networks (GANs)

A generative adversarial network consists of a generator and a discriminator, both neural networks. As the name suggests, the generator generates an image. It starts with pure noise and learns to evolve the noise into a plausible sample of the target class Y. Hence, it generates "fake" images indistinguishable from actual samples in the optimal case. On the other hand, the discriminator learns to differentiate precisely between the generated "fake" samples and the real data. The authors from [2] introduce an adversarial loss to implement the above. Let D be a discriminator network, and let G be a generator network, then the adversarial loss is given by:

$$\min_G \max_D V(D,G) = \mathbb{E}_x[\log D(x)] + \mathbb{E}_z[\log(1 - D(G(z))] \tag{1}$$

$D(x)$ denotes the probability that the discriminator predicts that x is a real sample. Consequently, $1 - D(G(z))$ denotes the probability with which the discriminators predict $G(z)$ to be a fake sample. Hence, the discriminator tries to maximize equation 1 while the generator tries to minimize it.

2.3 CycleGan

CycleGANs consist of two GANs connected in series. The first GAN transfers an image from $A \to B$ and the second GAN from $B \to A$. The idea is that when converting an image from one domain to the other and back again results in the same image, the network learns a mapping between the two domains. Note that to implement this characteristic, the adversarial losses of the GANs alone are insufficient. For example, the output of a generator can be a plausible sample of the target class without having any similarity to the input. Hence, the generator could always output the same image, indistinguishable from the target class, but without any correlation to the input image. Zhu et al. [3] introduce a cycle consistency loss to avoid this phenomenon and enable an actual style transfer. The cycle consistency loss (Equation 2) enforces that $F(G(x)) \approx x$ and $G(F(y)) \approx y$.

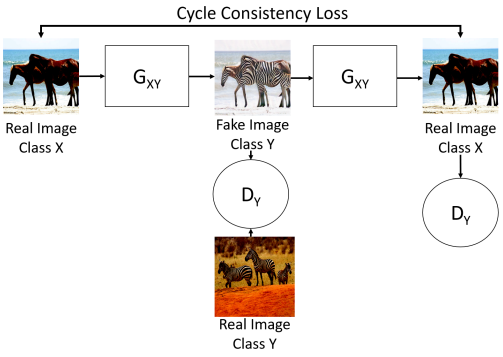


Figure 1: Architecture of a cycleGAN

$$L_{cyc}(G,F) = \mathbb{E}_x[||F(G(x)||_1] + \mathbb{E}_y[||G(F(y))||_1] \tag{2}$$

3 Data

For our experiments, we decided to use two datasets.The horse2zebra dataset from Berkeley's official directory of CycleGAN datasets [4] as a baseline and to validate our results with those from the original CycleGAN paper [3] and a Selfie2Anime Dataset [5], containing a collection of human and anime faces for our experiments and discovering limits of CycleGANs. Both datasets contain images of two distinct classes, providing no information as to which a sample from the one class would match with a representative from the other class. We call this type of data unpaired data (Figure 2).

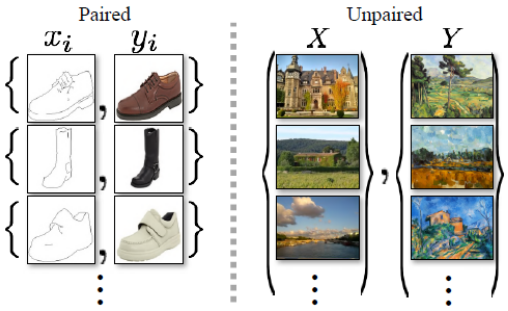


Figure 2: Paired training data (left) vs. unpaired training data right (src: [3])

The **horses2zebras** dataset is divided into a training and a test set. The train set contains 1067 horse and 1334 zebra images; the test set includes 120 horse images and 140 zebra images.

The **selfie2anime** dataset consists only of female photos. The training set comprised 3400 human selfies and 3400 anime face images from which we randomly selected 1700 each. The test set has 100 pictures from each class, respectively. Before training, all images are resized to 256 x 256 pixels and normalized to ensure that every pixel has a similar data distribution, allowing for faster convergence. Furthermore, we augment the data by horizontally flipping each image with a probability of 50% in every epoch.

4 Methods

4.1 Network Architecture

For the architecture of the CycleGAN, we closely followed the architecture described in [3], as implemented in PyTorch by the original authors, but also online tutorials and articles. Deviating from these tested approaches was not feasible, as the network architecture is quite complex and specific, and the network takes a long time to train, so changing the architecture or the implementation significantly would have taken a big amount of time.

The CycleGAN consists in fact of four different networks 3; two Generator networks which are used to transform images from domain X to domain Y, and vice versa, as well as two Discriminator networks, which try to classify images as real of generated, for each domain respectively. The Generator networks consist of an encoder part with three 2D convolutional blocks, followed by nine residual convolutional blocks, and a final decoder part consisting of three Transpose Convolution blocks. The final layer has the same size as the input and a hyberbolic tan function as an activation. The Discriminator networks are simpler, consisting of 5 convolutional layers with a leaky ReLU activation function. The final layer has a sigmoid activation function.

4.2 Loss

In order to train the CycleGAN, several different loss metrics have to be implemented. The **Discriminator Loss** is employed to determine the performance of the discriminator networks in the task of discerning real from generated data. In this case, the Mean Square Error (MSE) loss is calculated for the prediction of each discriminator for real and fake images. More specifically:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - y_i^*)^2 \tag{3}$$

where, y_i is the prediction of the discriminator for a given image, and y^* is an array of zeros (when the input image is generated) or an array of ones (when the input image is real). After the discriminator

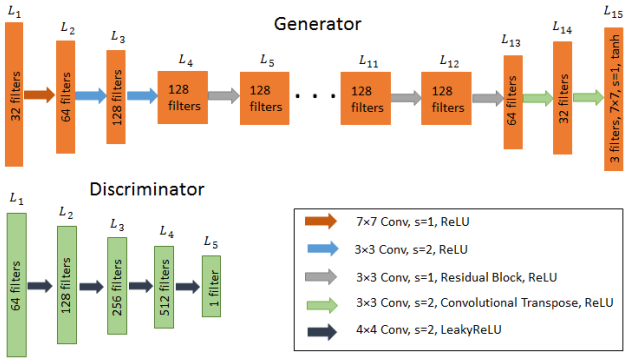


Figure 3: The Generator and Discriminator architecture of the CycleGAN (taken from [6]).

loss has been calculated for each discriminator, its mean is calculated and is used to backpropagate the error across both discriminator networks.

For the Generator networks, three loss metrics are employed. The **Generator Loss**, the **Cycle Loss** and the **Identity Loss**. The generator loss quantifies the quality of the generated data, on the basis of the prediction of the Discriminator networks. The MSE loss is calculated between the prediction of the discriminator network on the generated images, but in the case of the generator, the loss is maximized when the discriminator correctly identifies the images as fake (and minimized when the discriminator makes an incorrect prediction). Thus, the Discriminator and Generator networks are trained in competition with each other (thus the name Adversarial Networks).

The two types of losses that differentiate the CycleGAN from other types of GANs are the Cycle and Identity losses. The Cycle loss, as described above, quantifies the cycle consistency when a generated image is transformed back into its original domain. The loss is as follows:

$$CLOSS = \sum_{i=1}^{n} |y_{original} - y_{cycled}| \tag{4}$$

The Identity loss, in turn, is the L1 loss between the original and the transformed image, and it enforces structural similarity and color composition, which is useful for style transfer.

$$IDLOSS = \sum_{i=1}^{n} |y_{original} - y_{transformed}| \tag{5}$$

The total loss for both Generator networks is a weighted sum of all these components.

4.3 Extension: Using a semantic segmentation network to apply the style transfer on faces

After the results obtained from the CycleGAN were deemed sufficient, we used a pre-trained semantic segmentation model (specifically DeepLabv3 [7]), in order to only apply the style transfer on the face in each image (and not on the background). The way we achieved that is to input each original selfie through the segmentation network and obtaining a segmentation mask of the image. Subsequently, the mask was applied to the stylized image, to obtain the stylized face, which was combined with the background of the original image, and thus producing the final output.

5 Experiments

5.1 Horse2Zebra dataset - Baseline

As a baseline experiment, and in order to verify that our model is working as intended, we used the horse to zebra dataset (as described above). The hyperparameter setup for this experiment closely follows the one described in other implementations of the network that we could find. More specifically, the learning rate was $1e-4$, we trained for 100 epochs without using the identity loss, and the cycle loss was weighed as 10 times more important than the generator loss. Originally, there was a mistake in our implementation, specifically in the way the discriminator loss was setup, which made the Discriminator networks always predict the images as real, which in turn stalled the training of the Generator, resulting in very poor results (Fig. 9). After a lot of searching, the bug was identified, and we began our experiments.

Some of the results can be seen in Fig. 4. By visual inspection, the results on the test set are comparable to the ones obtained in the original CycleGAN paper. The model seems to perform relatively well on the horse to zebra transform, and more poorly on the reverse case. The network also fails sometimes when there are other objects present on the foreground of the image (e.g. humans), or when there are even surfaces. In these cases, the model adds stripes to elements of the image other than horses (Fig. 10. With the results looking similar to the ones we were aiming for, we continued with the selfies2anime dataset and our subsequent experiments.

5.2 Selfies2Anime dataset - Vanilla setup

The hyperparameter setup was the same as previously. We tried to increase the training time to 200 epochs, as the results contained a lot of artifacts when only training for 100 epochs. The improvement with the longer training time, was not significant. Curiously, the model seems to perform slightly better in this experiment when it comes to transforming anime faces to human ones (as it can be seen in Fig. 5). On the other hand, the fake anime faces contain a lot of artifacts (multiple eyes etc.). Our conclusion for this particular experimental set up was that the network could not perform the NST task consistently. The transformation between face domains seems to be a difficult one. Thus, further exploration of the training parameters was necessary.

5.3 Identity Loss, Loss weighting schemes

In the cycleGAN paper, the authors dispense of the use of an identity loss in the majority of their experiments. However, they stress that when generating photos from paintings, an identity loss can encourage the mapping to preserve color composition between the input and output [3]. In the vanilla version of the network, we realized that the hair color often changes between the original human selfie and the generated anime image (Figure 5a, 5b). Picking up on the idea from the paper, we included an identity loss in our computation and observed light bleaching in the hair color (Figure 6a, 6b). Furthermore, when we increased the impact of the identity loss by decreasing the weight factor for the cycle loss, our results improved significantly, and colors were preserved more often (Figure 7a,7b). However, while we perceive this effect as positive, the reverse translation was impaired. Increasing the impact of the identity loss shapes the generated human face more similar to the original anime faces, which look unnatural and uncanny to us.

5.4 Optimizer (Adam vs SGD + momentum)

The hyperparameter setup in this experiment was the same as in the vanilla setup, only that we now use stochastic gradient descent and momentum instead of Adam. The momentum term was set to 0.9 and the learning rate to 0.001. The obtained results from figure 13 show that the generator generates images that invert the colors, and sometimes it adds eyes in places where no eyes should be placed. Overall, the results indicate that the vanilla version in figure 5 outperforms this way of training the network.

5.5 Learning Rate Scheduling (Linear Decay vs. Cyclic Learning rates)

Another experiment that we tried was to compare different learning rate schedulers. We compared linear decay and cyclic learning rate decay using the vanilla setup, and trained for 100 epochs. From

visual inspection over the images, the cyclic learning rate scheduler performed much better than the linear learning rate scheduler. The latter had many artefacts, if it produced anything close to looking like an anime at all. The cyclic learning rate on the other had, did present a few good pictures (which are cherry-picked, however). See figure 11 for comparison.

5.6 Weight decay

In this experiment, we also implemented L2 weight regularization, through the Pytorch Adam optimizer. The weight decay λ term used for both The results from this experiment were of lower quality than the ones obtained without the weight regularization (Fig. 12). It can be argued that this is the case because the discriminator has to be powerful enough to discern generated from real inputs as effectively as possible, so the weight regularization interferes with the discriminator performance, thus making the training process slower.

5.7 Enhancing the results with face segmentation

As described above, we used a pre-trained semantic segmentation network in order to combine the stylized faces with the original background. Some of the resulting images can be viewed in Fig. 8. The segmentation network identifies the face area in most cases, except for when the image is very dark or there are objects that are obstructing the faces significantly. We think the effect of including the original background can be quite interesting is some cases.

6 Conclusion

Some of the conclusions that can be drawn from this line of work are:

- CycleGANs are, in general, capable of Neural Style Transfer given 2 sets of unpaired images from different domains. Even when the transform is quite complex (e.g. involves human faces), the results are still of sufficient quality under some conditions.
- The identity loss has a color-preserving effect. This consequence positively impacts the style transfer, from Selfies to Anime faces.
- Other modifications (changing the learning rate scheduling to cyclic, or changing the loss function weighting) did not have a conclusive effect.
- The weight regularization, changing the optimizer to SGD and using a linearly decaying learning rate all had a negative effect on the final results.

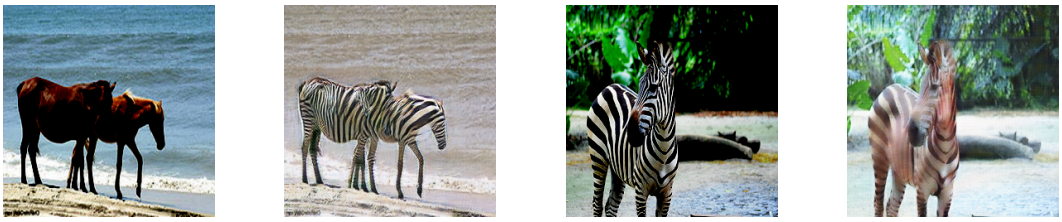
The future of NST is certainly interesting. In order to facilitate a more structured improvement in the field, one important suggestion would be to introduce a universal quantitative evaluation metric for stylized images. This would help with comparing different methods more effectively.

Appendix

A Code

Our code is available on this GitHub:
`https://github.com/simondoebele/dd2424-finalProject`
For implementing the code, we closely followed the implementation presented in this GitHub repository: `https://github.com/aladdinpersson/Machine-Learning-Collection`.

B Illustrations

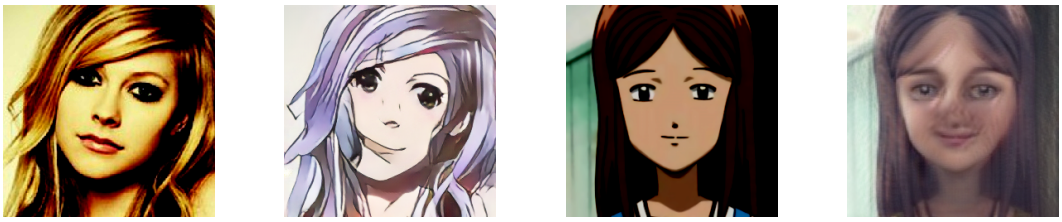


(a) Real Horse (b) Fake Zebra (c) Real Zebra (d) Fake Horse

Figure 4: horse2zebra - vanilla run

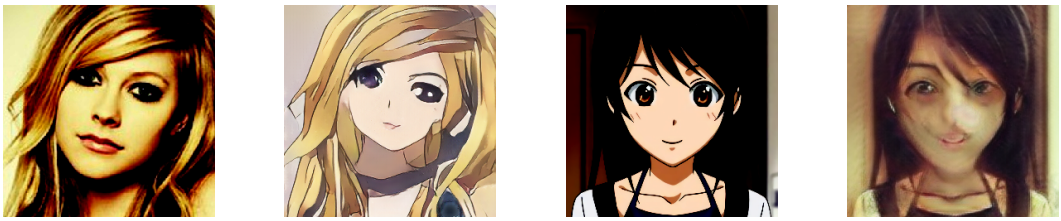
(a) Real Face (b) Fake Anime (c) Real Anime (d) Fake Face

Figure 5: selfie2anime - vanilla run, 200 epochs



(a) Real Face (b) Fake Anime (c) Real Anime (d) Fake Face

Figure 6: selfie2anime - including identity loss.

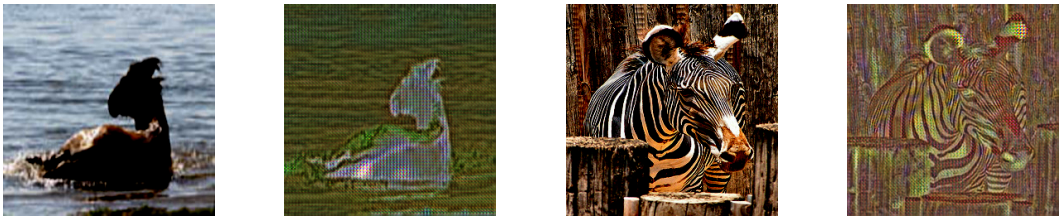


(a) Real Face (b) Fake Anime (c) Real Anime (d) Fake Face

Figure 7: selfie2anime - including identity loss and a lower weight factor for the cycle loss.

(a) Combined image (b) Combined image (c) Combined image (d) Combined image

Figure 8: selfie2anime - applying segmentation mask to combined stylized faces with original background.



(a) Real Horse (b) Fake Zebra (c) Real Zebra (d) Fake Horse

Figure 9: Poor results of our cycleCAN due to an implementation error in the training process.

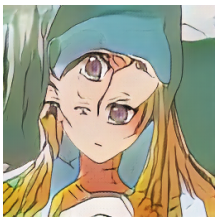
(a) Real Horse (b) Fake Zebra (c) Real Zebra (d) Fake Horse

Figure 10: Poor results showing some limitations of CycleGANs.

(a) Real Face (Cyclic Learning Rate)

(b) Fake Anime (Cyclic Learning Rate)

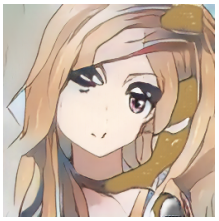
(c) Real Face (Linear Learning Rate Decay)



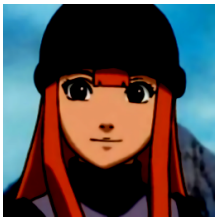
(d) Fake Anime (Linear Learning Rate Decay)

Figure 11: Results when using different learning rate schedulers.

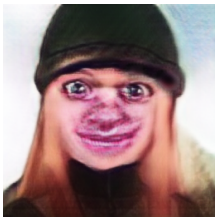
(a) Real Face



(b) Fake Anime



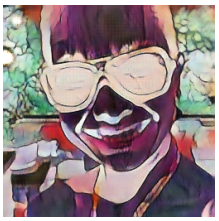
(c) Real Anime



(d) Fake Face

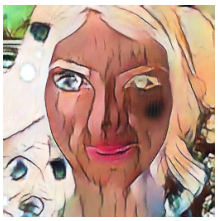
Figure 12: Results when using weight decay.

(a) Real Face



(b) Fake Anime

(c) Real Anime



(d) Fake Face

Figure 13: Results when using SGD and Momentum.

References

[1] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. A Neural Algorithm of Artistic Style. Technical Report arXiv:1508.06576, arXiv, September 2015. arXiv:1508.06576 [cs, q-bio] type: article.

[2] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks. Technical Report arXiv:1406.2661, arXiv, June 2014. arXiv:1406.2661 [cs, stat] type: article.

[3] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

[4] Berkeley UC. UC Berkeley's official directory of CycleGAN Datasets.

[5] Junho Kim. taki0112/UGATIT, May 2022. original-date: 2019-07-26T00:33:54Z.

[6] Muhaddisa Ali, Irene Gu, Mitchel Berger, Johan Pallud, Derek Southwell, Georg Widhalm, Alexandre Roux, Tomás Gomez Vecchio, and Asgeir Jakola. Domain mapping and deep learning from multiple mri clinical datasets for prediction of molecular subtypes in low grade gliomas. *Brain Sciences*, 10:463, 07 2020.

[7] Pytorch: Deeplabv3. `https://pytorch.org/hub/pytorch_vision_deeplabv3_resnet101/`. Accessed: 2022-05-25.

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[9] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[10] Philip TG Jackson, Amir Atapour Abarghouei, Stephen Bonner, Toby P Breckon, and Boguslaw Obara. Style augmentation: data augmentation via style randomization. In *CVPR Workshops*, volume 6, pages 10–11, 2019.