CycleGAN for Emojis

Sofia Ek **Daniel Gedon** **Carmen Lee**

Department of Information Technology
Uppsala University
Uppsala, Sweden

Abstract

Cycle-Consistent Adversarial Networks (CycleGAN) describe a specific training method where a general GAN loss is combined with a cycle consistency loss. The method can be used to learn style transfer between two visual domains when paired images are not available. In this project, we implemented a small-scale architecture for CycleGAN and tested it on the transfer between emojis of Apple and Windows styles.

1 Introduction

Image-to-image translation is a set of problems where the goal is to modify an input image by generating a synthetic version of it in a predefined way. Examples include translating a painted image into a rendered photography or a landscape in daylight to sunset. Traditionally this requires a large dataset of aligned image pairs between the two domains/styles. The paired image set is problem specific and usually expensive, or in some scenarios, not even possible to obtain.

The Cycle-Consistent Adversarial Network (CycleGAN) [Zhu et al., 2017] is a technique for image-to-image translation without the need for paired images. The network is instead trained with two separate sets of images, one from the input domain and another from the output domain. On top of the general GAN loss, the model uses the idea of cycle consistency: when an input image is transformed by the generative network to an image in the output domain and this image is transformed back to the input domain, the resulting image should be equal or as close as possible to the original image in some measure. In this way, CycleGAN is able to train the model without paired images.

In this project we implement a CycleGAN that can translate between Apple emojis and Windows emojis. The dataset and architecture are adopted from the course [Grosse, 2018]. The code is publicly available at `github.com/dgedon/CycleGAN-for-Emojis`.

2 Related Work

The CycleGAN method builds on the "pix2pix" model [Isola et al., 2017], which requires a training set of paired images to learn the mapping from input to output domain. It is based on a conditional generative adversarial network [Mirza and Osindero, 2014, Goodfellow et al., 2014] that can generate new images given a training dataset.

Another method [Liu et al., 2017], developed at the same time as CycleGAN, can similarly handle a training set of unpaired images. It is also based on conditional generative adversarial networks but in combination with variational autoencoders [Kingma and Welling, 2013].

Preprint. Under review.

3 Methods

Generative adversarial network (GAN) has two components, a generator, denoted ad $D(x; \theta_d)$ and a discriminator, denoted as $G(y; \theta_g)$. A generator maps $x \in X$ to $y \in Y$ and a discriminator trains to distinguish whether the generated output is a true member of the distribution $p_{\text{data}}(y)$. The GAN objective is therefore

$$\min_G \max_D V(D, G) = \mathbb{E}_{y \sim p_{\text{data}}(y)}[\log D(y)] + \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log(1 - D(G(x))], \tag{1}$$

where D tries to maximize the value function V while G tries to minimize it.

While CycleGAN [Zhu et al., 2017] also trains a mapping $G : X \to Y$ such that real images from the target domain $y \in Y$ cannot be distinguished from the generated output $\hat{y} = G(x)$, $x \in X$, it does so without paired-images. The network is trained given training samples from both a source set $\{x_i\}_{i=1}^N$, $x_i \in X$ and a target set $\{y_j\}_{j=1}^M$, $y_j \in Y$. Compared to a vanilla GAN [Goodfellow et al., 2014], the loss function is modified to directly use the output of the discriminator instead of the $\log$ of it for improved stability:

$$\min_G \mathcal{L}_{\text{GAN}}(G, D, X, Y) = \mathbb{E}_{x \sim p_{\text{data}}(x)}[(1 - D(G(x))^2] \tag{2}$$

$$\min_D \mathcal{L}_{\text{GAN}}(G, D, X, Y) = \mathbb{E}_{y \sim p_{\text{data}}(y)}[(1 - D(y))^2] + \mathbb{E}_{x \sim p_{\text{data}}(x)}[D(G(x)^2] \tag{3}$$

This results in a network that learns the empirical distribution $p_{\text{data}}(y)$, but there is no guarantee that individual images are translated in a useful way. To enforce a meaningful mapping for individual samples a second generative network is trained for the inverse mapping $F : Y \to X$. These two generative networks G and F are coupled with a cyclic consistency loss which enforces $F(G(x)) \approx x$ and $G(F(y)) \approx y$, respectively. The mapping process and the cyclic consistency loss are illustrated in Fig. 1.

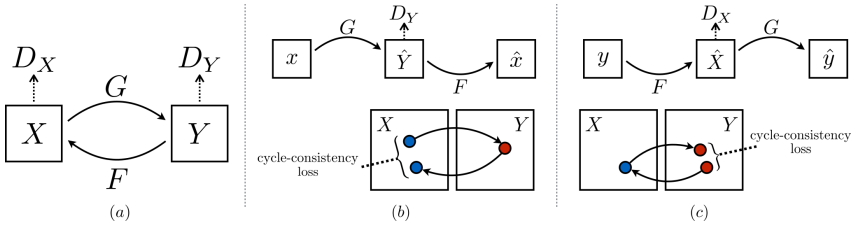


Figure 1: The mapping functions and the cyclic consistency loss. Image taken from [Zhu et al., 2017].

The combined cyclic consistency loss for the forward and backward direction is defined as the distance between the original image and the resulting image after the forward and backward transformations:

$$\mathcal{L}_{\text{cyc}}(G, F) = \mathbb{E}_{x \sim p_{data}(x)}[||F(G(x)) - x||_1] + \mathbb{E}_{y \sim p_{data}(y)}[||G(F(y)) - y||_1] \tag{4}$$

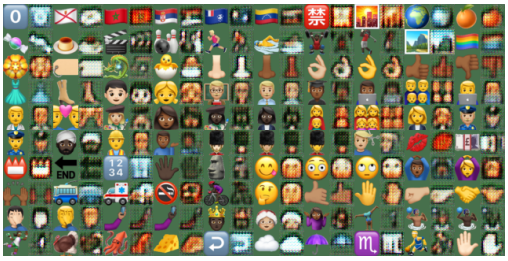
The full loss includes the terms of GAN losses when each set is used as the source set and the other as the target set, as well as the cyclic consistency losses scaled by a factor λ:

$$\mathcal{L}(G, F, D_X, D_y) = \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) + \mathcal{L}_{\text{GAN}}(F, D_X, Y, X) + \lambda \mathcal{L}_{\text{cyc}}(G, F) \tag{5}$$

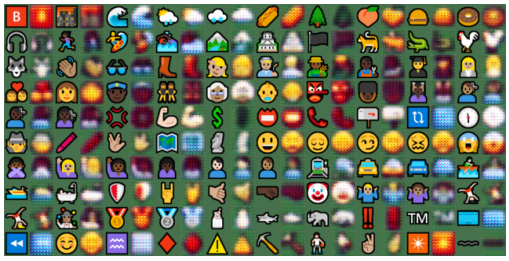
4 Data, Experiments and Findings

We use an emoji dataset containing Apple-style and Windows-style emojis. The dataset is designed for a course assignment in [Grosse, 2018]. The choice of the dataset is appropriate because the two sets have similar image resolutions and themes but distinctive styles. There are 2370 Apple-style and 2114 Windows-style emojis. Out of each set, 100 emojis are selected for testing. Since the images are quite simple, only basic preprocessing were performed to make sure that the images are of the same size (resized to 32×32) and are normalized.

We use a small-scale architecture in this project, since the image sizes and the datasets are fairly small. The CycleGAN is constructed with two discriminators and two generators that are trained simultaneously.



(a) Apple to Windows-style after 1 epoch.



(b) Windows to Apple-style after 1 epoch.

(c) Apple to Windows-style after 220 epoch.

(d) Windows to Apple-style after 220 epoch.

Figure 2: Emojis and their style-transferred counterparts. Top row: illustration after one epoch. Bottom row: illustration after 220 epochs. Left column: Paris of Apple to Windows style. Right column: Paris of Windows to Apple style.

The two discriminators use the same architecture. They consist of a convolution neural network with four layers. We use a kernel size $K = 4$, stride 2 and zero padding 1 for the first three layers. Batch norm is used after each convolution and rectified linear unit (ReLU) is used as the activation function. For the last layer, no padding is used and the activation function is the Sigmoid function. We use filter sizes of {32, 64, 128, 1} for the four layers.

The two generators also use the same architecture. It is a neural network with two convolution layers for downsampling, one residual block, and two deconvolution layers for upsampling. We use a kernel size $K = 4$, stride 2 and zero padding 1 for the convolution and deconvolution layers. The residual block uses a kernel size $K = 3$, stride 1 and zero padding 1 to keep the dimensions. Batch norm is used after each convolution and ReLU is used as activation function for all layers except the last one. There, the number of output channels are back to 3 and the activation function is the tanh function. We use filter sizes of {32, 64} for the convolutions, 64 for the ResNet block and {32, 3} for the deconvolutions. Note that we initialized all convolutional layers with as Gaussian with zero mean and standard deviation of 0.001.

We use the batch size 16, learning rate of $3 \cdot 10^{-4}$, and parameters $\beta = [0.5, 0.999]$ for the Adam optimizer. The scaling factor λ for the cyclic consistency loss is set to 2. The network is trained for 500 epochs. Since the two datasets for Windows and Apple emojis are of different sizes, we randomly select 2014 Apple emojis in each epoch to match the number of Windows training emojis.

We measure both the generator loss and the discriminator loss. They indicate how well the generator and the discriminator performs respectively. We also visually inspect the test samples and their style-transferred counterparts every 20 epochs, see Fig. 2 for generated images after one epoch and 220 epochs which yield the highest quality images.

5 Challenges and Conclusion

The main challenge for the reimplementation was the choice of architecture. The original architecture in [Zhu et al., 2017] suits for their types of experiments with larger sized images. We constrain ourselves for computational reasons to smaller sized images and therefore need to adapt the architecture accordingly. We follow the same architectural guidelines as described in the course of [Grosse, 2018] by checking the accompanying assignment description and code. Further, we tested and optimized

some hyperparameters (batch size, λ for the cycle consistency loss scaling factor, learning rate). We concluded that the values given in the course assignment were already optimal.

One crucial part to get high quality images was the correct initialization of the images. First, we initialized the network using the standard PyTorch network initialization [Glorot and Bengio, 2010]. However, this resulted in very rough grained images, with chessboard like patterns. Considering the same initialization as in the paper (or the course assignment) yielded the resulting high quality images.

A second important part is to control for overfitting. We observed that when we train for too long, the style transfer between domains becomes more of a copy from input to output. This could probably be alleviated with specific regularization schemes.

The key takeaway from this reimplementation is the adaptation of the architecture to the given dataset. This is naturally accompanied by an architectural dependent hyperparameter search for the optimization procedure.

6 Ethical consideration and societal impact

Generative adversarial networks, and more specifically CycleGANs, can produce or translate images into realistic new ones. This could in most scenarios be used as a powerful tool to improve images or to generate samples when data are sparse. The method, however, can also be used to create fake images for malicious purposes. There exist many tutorials on how to setup and train these networks and pre-trained networks are commonly accessible. This easy access means that the technology can easily be misused if such intent exists. Prior to deep learning, this type of image editing could be done manually with a graphic editor, but the new technique allows for faster manipulation of many images at the same time and such extensions are possible for video manipulation. The positive side is that such techniques are not yet mature enough to pass as authentic. However, it is in the foreseeable future that we will not be able to distinguish between real and fake images/videos. As a result, social awareness needs to be increased and legislation must be put in place to regulate the use of such techniques.

References

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.

Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.

Roger Grosse. Intro to neural networks and machine learning, 2018. URL `http://www.cs.toronto.edu/~rgrosse/courses/csc321_2018/`.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. *arXiv preprint arXiv:1703.00848*, 2017.

Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

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.