The 33rd Annual Conference of the Japanese Society for Artificial Intelligence, 2019

Cycle Sketch GAN: Unpaired Sketch to Sketch Translation Based on Cycle GAN Algorithm

Takeshi Kojima

Peach Aviation Limited

Unlike pixel image generation, sketch drawing generative model outputs a sequence of pen stroke information. This paper proposes Cycle Sketch GAN: the first model that learns to translate a sketch drawing from source domain to target domain in the absence of paired dataset. Based on Cycle GAN algorithm, this model uses Transformer Encoder architecture in generators. Transformer Encoder feeds the input stroke information in source domain and generates the parameters for output distribution, from which the stroke information in target domain is sampled by reparameterization trick. The negative log likelihood of the distribution is used as cycle consistency loss. This model is trained and evaluated by some QuickDraw datasets. Qualitative evaluation shows that this model can practically translate sketch drawings from source domain to target domain. Quantitative evaluation by user study showed that 42 % of the translated sketches is recognizable compared to 71 % of the human sketches.

1. Introduction

Unlike pixel image generation, sketch drawing generative model outputs a sequence of pen stroke information (See section 2.1 for details). Some recent researches focused on creating sketch drawing generative model [Ha 18][Song 18] by neural network. However, the research of sketch to sketch translation, which aims to transform a sketch from source domain to target domain especially without supervised dataset, was not yet conducted.

This paper proposes Cycle Sketch GAN: the first model that learns to translate a sketch drawing from source domain to target domain in the absence of paired dataset. Specifically, this unsupervised learning model can change a sketch drawing's partial shapes characteristic to source domain into ones characteristic to target domain while keeping the common features unchanged (See an example in Fig.1). To train this model, we need to prepare 2 domain datasets, but each data in one domain does not need to have paired data in the other domain. This model is based on Cycle GAN algorithm[Zhu 17]. However, several changes are implemented to solve the following 2 problems specific to sketch drawing process.

The first problem of unsupervised sketch to sketch translation is that the sketch drawing is a process of generating a sequence of stroke vector representations. Therefore, Convolutional Neural Network(CNN), which is used in basic Cycle GAN, should not be used for this solution. Cycle Sketch GAN solves this problem by using Transformer[Vaswani 17] Encoder architecture in a generator. Transformer has self-attention mechanisms and recently succeeded in improvements of several sequential data processing tasks mainly in NLP. Note that Transformder Decoder is not used in this paper because we have no supervised data. Instead, Transformer Encoder feeds input and directly generates sequential outputs.

The second problem is that drawing sketch requires the accurate generation of strokes. Therefore, cycle consistency loss function has to be like the form of reconstruction loss as in [Ha 18], instead of L1 or L2 Norm. Cycle Sketch GAN solves this problem as follows: Transformer Encoder feeds the input stroke in source domain and generates the parameters for output distributions. Stroke information in target domain is sampled from the distribution by reparameterization trick to enable backpropagation for the training. The negative log likelihood of the distribution is used as cycle consistency loss to improve accuracy.

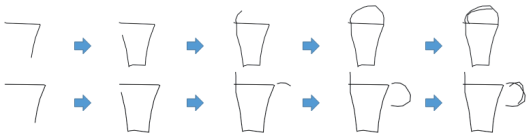

Figure 1: A visualized example of sequential stroke output process by Cycle Sketch GAN. This model can learn to translate a sketch drawing from source domain (Top: bucket) to target domain (Bottom: cup) in the absence of paired datasets. The common feature between 2 drawings (the body of bucket and cup) is unchanged.

2. Methods

2.1 Data Format

Based on [Ha 18], the data format of a sketch drawing for this model is a sequence of pen stroke actions $s = (s1, ...s_i, ..., s_{N_{max_s}})$, where $s_i = (\Delta_{x_i}, \Delta_{y_i}, p1_i, p2_i, p3_i)$. $\Delta_{x_i}, \Delta_{y_i}$ are the offset distance of ith pen movement in the direction of x axis and y axis. $p1_i, p2_i, p3_i$ are binary one-hot vector of 3 possible states at ith movement[*1]. $p1_i$ is an indicator that the pen is touching the paper for the ith pen movement. $p2_i$ is an indicator that the pen is lifted from the paper for the ith pen movement. $p3_i$ is an indicator that the drawing has ended. In case of $p3_i = 1$, Δ_{x_i} and Δ_{y_i} are defined to be 0.

2.2 Generator Architecture

This model uses Transformer[Vaswani 17] Encoder architecture as a generator to translate a sketch drawing stroke representation of domain S_A to domain S_B, and vice versa. Specifically, $s_A \in S_A$ is fed into one generator and translated to $\tilde{s_B} \in S_B$. In the same way, $s_B \in S_B$ is fed into the other generator and translated to $\tilde{s_A} \in S_A$. The architectures of these 2 generators are the same. This section omits the subscript A and B for simplicity.

Contact: t.kojima@flypeach.com

*1 [Ha 18] defines $p1_i, p2_i, p3_i$ as binary one-hot vector of 3 possible states at $i + 1$th movement.

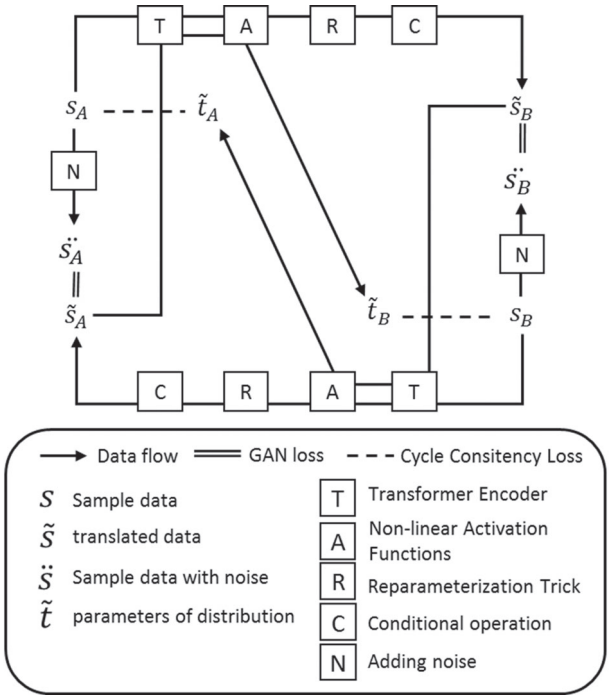

Figure 2: Overall image of Cycle Sketch GAN

The input s is concatenated with Positional Encoding [Vaswani 17] whose dimension size for ith position is ${N_{PE}}^{*2}$. The concatenated representation is fed into Transformer Encoder as input. Transformer Encoder has N_{TL}multiple stacked layers, each of which consists of N_{TA} multi-head attentions and a feed-forward network with dimension size of $4N_{TA}$, followed by a position-wise linear projection for the output below.

$$\begin{aligned}&(\mu_x, \mu_y, \hat{\sigma_x}, \hat{\sigma_y}, \hat{\rho_{xy}}, q1, q2, q3)_1 \\ &\quad .... \ (\mu_x, \mu_y, \hat{\sigma_x}, \hat{\sigma_y}, \hat{\rho_{xy}}, q1, q2, q3)_{N_{max_s}} \\ &\quad = TransformerEncoder([s; PE]) \end{aligned} \quad (1)$$

As described in [Ha 18], $(\mu_x, \mu_y, \hat{\sigma_x}, \hat{\sigma_y}, \hat{\rho_{xy}})$ are defined as the parameters for a bivariate normal distribution to describe Δx and Δy. $(q1, q2, q3)$ are the categorical distribution parameters to model the ground truth data $(p1, p2, p3)$. The following nonlinear functions are required to ensure that standard deviations are non-negative, that the correlation values are limited between -1 and 1.

$$\sigma_x = exp(\hat{\sigma_x}), \ \sigma_y = exp(\hat{\sigma_y}), \ \rho_{xy} = tanh(\hat{\rho_{xy}}) \quad (2)$$

The reparameterization trick for bivariate normal distribution is applied using $(\mu_x, \mu_y, \sigma_x, \sigma_y, \rho_{xy})$ to sample single value for each ith pen movements.

$$\begin{aligned}\begin{bmatrix} \tilde{\Delta x_i} \\ \tilde{\Delta y_i} \end{bmatrix} &= \begin{bmatrix} \mu_{x_i} \\ \mu_{y_i} \end{bmatrix} + L_i \begin{bmatrix} \epsilon_{x_i} \\ \epsilon_{y_i} \end{bmatrix} \\ where \ \ L_i &= \begin{bmatrix} \sigma_{x_i} & 0 \\ \rho_{xy_i}\sigma_{y_i} & \sqrt{1-{\rho_{xy_i}}^2}\sigma_{y_i} \end{bmatrix} \\ \epsilon_{x_i}, \epsilon_{y_i} &\sim N(0,1), \ \epsilon_{x_i}, \epsilon_{y_i} \in \mathbb{R} \end{aligned} \quad (3)$$

L_i is a lower triangular matrix after cholesky decomposition of covariance matrix of bivariate normal distribution.

∗2 Considering the case $N_{PE} > 5$, we use concatenation instead of addition.

Categorical reparameterization for $(q1, q2, q3)_i = q_i$ is also applied by using Gumbel Softmax[Jang 17] with temperature τ.

$$\begin{aligned}\tilde{q_i} &= softmax((q_i + g_i)/\tau) \\ where \ \ g_i &= -\log(-\log(u_i)) \\ u_i &\sim Uniform(0,1), \ u_i \in \mathbb{R}^3 \end{aligned} \quad (4)$$

In order to deceive discriminators as much as possible, generators calculate the following position-wise conditional operation before the data is fed into Discriminator.

$$\tilde{s_i} = \begin{cases} (0, 0, \tilde{q1}, \tilde{q2}, \tilde{q3})_i & if \ max(\tilde{q1}_i, \tilde{q2}_i, \tilde{q3}_i) = \tilde{q3}_i \\ (\tilde{\Delta x}, \tilde{\Delta y}, \tilde{q1}, \tilde{q2}, \tilde{q3})_i & otherwise \end{cases} \quad (5)$$

Here, for simplification, the sequence of functions (1) and (2) can be defined as $t = (t1, ..., t_i, ..., t_{N_{max_s}}) = G(s)$, where $t_i = (\mu_x, \mu_y, \sigma_x, \sigma_y, \rho_{xy}, q1, q2, q3)_i$. The sequence of functions (3), (4) and (5) can also be defined as $\tilde{s} = (\tilde{s}_1, \tilde{s}_2, ..., \tilde{s}_i, ..., \tilde{s}_{N_{max_s}}) = H(t)$. By using these expressions, 2 generators can be defined as:

$$\tilde{s_B} = H_B(G_B(s_A)), \ \ \tilde{s_A} = H_A(G_A(s_B)) \quad (6)$$

Note that the function $H_A(\cdot)$, $H_B(\cdot)$ does not contain any neural networks to be optimized.

2.3 Discriminator Architecture

The discriminator D is a multilayer convolutional neural network with instance normalization. Specifically, the discriminator regards the input s or $\tilde{s}$ as an image with height=N_{max_s}, width=5 and depth=1. The jth layer of D convolves the input of the layer with kernel size $(N_{Dk-height_j}, \ N_{Dk-weight_j})$ and stride size = N_{Ds_j} without padding, and outputs the tensor with channel size N_{Dc_j}, followed by instance normalization and activation. As an exception, the first layer and final layer does not have instance normalization. The final layer also has no activation function due to the restriction of the architecture of the discriminator in improved Wasserstein GAN optimization process [Gulrajani 17]. 2 discriminators, D_A for data A and D_B for data B are implemented with the same architecture described above.

2.4 Objective function

The objective function of this model contains adversarial losses and cycle consistency losses for both domains[Zhu 17]. As for the adversarial loss, this model uses improved Wasserstein GAN [Gulrajani 17] instead of the normal GAN [Goodfellow 14] to avoid mode collapse and to stabilize the training. The adversarial loss for data A is:

$$\begin{aligned}&\mathcal{L}^A_{GAN}(G_A, D_A, S_B, S_A) \\ &\quad = \mathbb{E}_{s_B}[D_A(H_A(G_A(s_B)))] - \mathbb{E}_{\ddot{s}_A}[D_A(\ddot{s}_A)] \\ &\qquad + \textit{gradient penalty for WGAN-GP} \\ &where \ \ddot{s}_A = noise(s_A) \end{aligned} \quad (7)$$

$noise(\cdot)$ is a function that adds random noises $U(-0.01, 0.01)$ into $(p1, p2, p3)$, and also into $(\Delta_{x_i}, \Delta_{y_i})$ if $q3_i = 1$ to prevent D from concentrating too much on discrete variables. The same as true for the adversarial loss of data B, $\mathcal{L}^B_{GAN}(G_B, D_B, S_A, S_B)$.

As for cycle consistency loss of data A, firstly the following cycled parameters are introduced.

$$(\tilde{t}_{A,1}, ..., \tilde{t}_{A,i}, ...\tilde{t}_{A,N_{max_s}}) = G_A(H_B(G_B(s_A)))$$
$$where\ \tilde{t}_{A,i} = (\tilde{\mu}_x^A, \tilde{\mu}_y^A, \tilde{\sigma}_x^A, \tilde{\sigma}_y^A, \tilde{\rho}_{xy}^A, \tilde{q1}^A, \tilde{q2}^A, \tilde{q3}^A)_i \quad (8)$$

Then, cycle consistency loss for data A is expressed as follows:

$$\mathcal{L}_{cyc}^A(G_A, G_B, S_A)$$
$$= \mathbb{E}_{s_A}\Big[- \frac{1}{N_{\max_s}} \sum_{i=1}^{N_s} \log\Big(\mathcal{N}(\Delta x_i^A, \Delta y_i^A \mid \tilde{w}_i^A)\Big)$$
$$- \frac{1}{N_{\max_s}} \sum_{i=1}^{N_{\max_s}} \sum_{k=1}^{3} pk_i^A \log(qk_i^{'A})\Big]$$
$$where\ \tilde{w}_i^A = (\tilde{\mu}_x^A, \tilde{\mu}_y^A, \tilde{\sigma}_x^A, \tilde{\sigma}_y^A, \tilde{\rho}_{xy}^A)_i$$
$$q1_i^{'A}, q2_i^{'A}, q3_i^{'A} = softmax(\tilde{q1}_i^A, \tilde{q2}_i^A, \tilde{q3}_i^A) \quad (9)$$

$\mathcal{N}(\Delta x_i^A, \Delta y_i^A \mid \tilde{w}_i^A)$ is the probability distribution function for a bivariate normal distribution. N_s is the point of last stroke in the sketch. This cycle consistency loss function is the same as the reconstruction loss function of [Ha 18] except that the distribution of $(\Delta x, \Delta y)$ is not modeled as a Gaussian mixture model (GMM). It can be said that the function is a special case of GMM size = 1. The same as true for the cycle consistency loss for data B, $\mathcal{L}_{cyc}^B(G_B, G_A, S_B)$.

The final objective function is:

$$\begin{aligned}\mathcal{L}(G_A, G_B, D_A, D_B) &= \mathcal{L}_{GAN}^A(G_A, D_A, S_B, S_A) \\ &+ \mathcal{L}_{GAN}^B(G_B, D_B, S_A, S_B) \\ &+ \lambda\mathcal{L}_{cyc}^A(G_A, G_B, S_A) \\ &+ \lambda\mathcal{L}_{cyc}^B(G_B, G_A, S_B) \quad (10)\end{aligned}$$

We aim to solve:

$$G_A^*, G_B^* = \arg\min_{G_A, G_B} \max_{D_A, D_B} \mathcal{L}(G_A, G_B, D_A, D_B) \quad (11)$$

3. Experimtents

3.1 Dataset

To evaluate Cycle Sketch GAN, full size of "Sketch-RNN QuickDraw Dataset" is used. QuickDraw Dataset contains hundreds of classes of sketch drawings. Each class is a dataset of more than 70K training samples and 2.5K test samples. In this paper, the following 6 classes are picked up from QuickDraw and made pairs: (bucket, cup), (suitcase, envelope), (sock, rollerskates). Note that each data in one class does not have paired data in the other class. The data format is changed according to section 2.1.

This experiment only uses the data whose size of the stroke actions does not exceed $N_{max_s} = 50$. Moreover, in order to equalize the training dataset size between paired classes, training data were randomly sampled from the class whose dataset size is larger than the other.

3.2 Implementation details

As for generators, the dimension size of Positional Encoding is set to be $N_{PE} = 251$. Transformer layer size is $N_{TL} = 12$, and the multihead attention size is $N_{TA} = 4$. As for the Discriminator, the hyper parameter values are set as followed:
$(N_{Dk-height_1}, N_{Dk-weight_1}, N_{Ds_1}, N_{Dc_1}) = (5, 5, 1, 128)$
$(N_{Dk-height_2}, N_{Dk-weight_2}, N_{Ds_2}, N_{Dc_2}) = (10, 1, 2, 256)$
$(N_{Dk-height_3}, N_{Dk-weight_3}, N_{Ds_3}, N_{Dc_3}) = (10, 1, 2, 512)$
$(N_{Dk-height_4}, N_{Dk-weight_4}, N_{Ds_4}, N_{Dc_4}) = (5, 1, 1, 1)$.

Figure 3: The original sketches by human (Row A, B) and the translated sketches by Cycle Sketch GAN (Row B', A').

All the activation functions in generators and discriminators are Leaky-ReLU activation. The temperature τ of Gumbel Softmax is set to be 1. λ in the final objective function is also set to be 1.

While training, minibatch SGD is used and the Adam optimizer is applied for the optimization process with $\beta_1 = 0$ and $\beta_2 = 0.9$ [Gulrajani 17]. The learning rate is fixed with 0.00005 during the training. The minibatch size is 128 and the iteration size is 30000 for generators. Discriminators and generators are mutually trained with the number of iterations of Discriminators per iteration of generators be set 5[Gulrajani 17].

Before generators feed data, the offsets $(\Delta x, \Delta y)$ in each classes is normalized using a single scaling factor [Ha 18] to adjust the offsets in the training set to have a standard deviation of 1. The test dataset is also normalized by that single scaling factor which is calculated by the training set of the same class.

2 types of data augmentation are applied into the training data for every iteration[Ha 18]. The first one is to stretch Δx and Δy respectively by multiplying random value drawn from $U(0.85, 1.15)$. The second one is to drop out strokes by dropping each point within line segments with a probability of 0.1.

Residual Dropout[Vaswani 17] is applied into Transformer Encoder with dropout rate = 0.1 during the training. At inference time, the dropout rate = 0, and also $\epsilon_x, \epsilon_y, g = 0$ in the reparameterization trick to produce the optimal translated drawings $\tilde{s_B}^* = H_B(G_B^*(s_A))$ and $\tilde{s_A}^* = H_A(G_A^*(s_B))$.

3.3 Results

3.3.1 Qualitative Evaluation

3 Cycle Sketch GAN models are trained by using 3 dataset pairs in QuickDraw (See section 3.1). Fig. 3 shows some examples of the translated sketch drawings from test data by the trained models. A and B rows are the randomly chosen sample drawings from each class test datasets. The drawings in B' rows are respectively the translated sketch drawings from the above A drawings. The

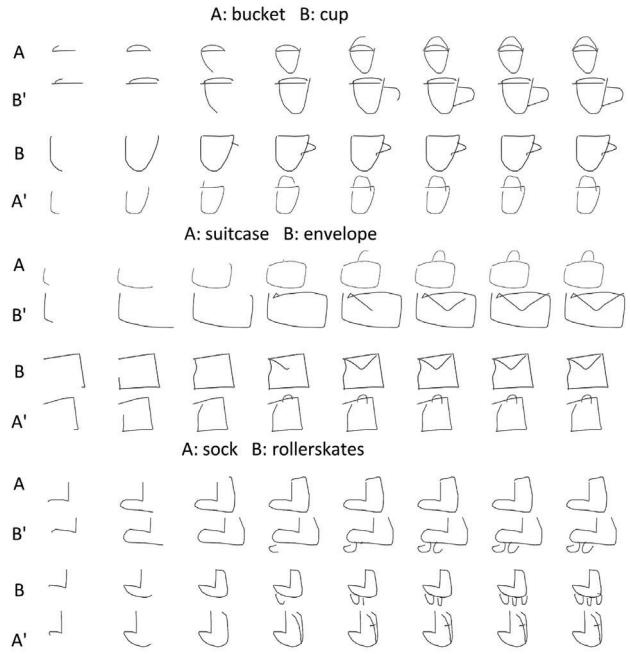

Figure 4: The stroke order of an original sketch by human (A, B) and the translated sketch by Cycle Sketch GAN (B', A').

same is true for A' and B. It shows that the common features are unchanged between the test sample drawing and the translated drawing, but the partial shapes characteristic to source domain are successfully transformed into ones characteristic to target domain, although some samples are failed to be translated.

Fig. 4 compares the order of the strokes between some test sample data and the translated data. Each row describes the snapshots of the strokes at the point of $i = 6, 12, 18, 24, 30, 36, 42, 48$ from the left to right. It clearly shows that the stroke order between original sketch and translated sketch are very similar from the start until some point. After that point, the stroke becomes different to express the characteristic feature for each domains.

3.3.2 Quantitative Evaluation

Class	Cycle Sketch GAN	Human
bucket	32.8($\pm$30.5)	57.6($\pm$25.9)
cup	43.0($\pm$28.5)	75.2($\pm$31.3)
suitcase	49.0($\pm$30.3)	54.4($\pm$27.0)
envelope	63.1($\pm$23.1)	85.8($\pm$18.7)
sock	38.4($\pm$26.1)	71.8($\pm$23.9)
rollerskates	30.8($\pm$14.8)	82.0($\pm$14.7)
Average	42.9	71.1

Table 1: User survey result

User study was also conducted on Amazon Mechanical Turk (AMT) to test the quality of translated sketch drawings. For each class, the survey results were collected from 20 participants. Specifically, participants see drawings one by one, and were asked "Do you think the drawing is (class name) ? ". Participants clicked on "yes" or "no" for the answer. For each dataset class, 55 drawings are surveyed, which consists of 25 of real sample sketches by humans, 25 of translated sketches by Cycle Sketch GAN, and 5 of trials (apparently irrelevant class sketches to check whether the participant's response was reliable or not). The order of showing these drawings are randomized.

Table. 1 shows the survey result, which lists the average percentage (and the standard deviation) of answering "yes" for each class surveys. Overall, 42 % of the translated sketches is recognizable compared to 71 % of the human sketch. There is apparent that some easy sketches such as envelope got higher recognition rate, while some complex sketches such as rollerskates got lower rate.

4. Conclusion and Future Work

This paper proposed Cycle Sketch GAN, the first model that learns to translate a sketch drawing from source domain to target domain in the absence of paired dataset. Qualitative and quantitative evaluation by some QuickDraw datasets demonstrates the effectiveness of this model. As a future work, the model needs to be improved to be able to translate complicated sketch drawings with higher quality. Using GMM as output distributions, or using Encoder-Decoder architecture as generators such as Seq2Seq might be effective. Furthermore, there might be a possibility that this unsupervised learning approach can be applied into other tasks that requires sequential data generation, such as unsupervised language translation[Lample 18], even though we need quite a lot of model changes and improvements.

References

[Goodfellow 14] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y.: Generative Adversarial Nets, in *Advances in Neural Information Processing Systems 27*, pp. 2672–2680 (2014)

[Gulrajani 17] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C.: Improved Training of Wasserstein GANs, in *Advances in Neural Information Processing Systems 30*, pp. 5767–5777 (2017)

[Ha 18] Ha, D. and Eck, D.: A Neural Representation of Sketch Drawings, in *International Conference on Learning Representations* (2018)

[Jang 17] Jang, E., Gu, S., and Poole, B.: Categorical Reparameterization with Gumbel-Softmax (2017)

[Lample 18] Lample, G., Ott, M., Conneau, A., Denoyer, L., and Ranzato, M.: Phrase-Based & Neural Unsupervised Machine Translation, in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 5039–5049 (2018)

[Song 18] Song, J., Pang, K., Song, Y., Xiang, T., and Hospedales, T. M.: Learning to Sketch With Shortcut Cycle Consistency, in *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018*, pp. 801–810 (2018)

[Vaswani 17] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I.: Attention is All you Need, in *Advances in Neural Information Processing Systems 30*, pp. 5998–6008 (2017)

[Zhu 17] Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A.: Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks, in *Computer Vision (ICCV), 2017 IEEE International Conference on* (2017)