cGANs for Cartoon to Real-life Images

Kanya Satis	Pranjal Singh Rajput	Sonnya Dellarosa	Wenxuan Huang	Obinna Agba
4777336	4788087	4783344	4925777	4793765

Delft University of Technology
CS4180 Deep Learning
Group 25

1. Introduction

Image-to-image translation is a learning task to establish visual mapping between an input and output image. The task has several variations differentiated based on the purpose of the translation, such as synthetic→real translation [17][21], photo→caricature translation [23] and many others. The problem has been tackled using different approaches, either through traditional computer vision methods [7], as well as deep learning approaches in recent trends.

One approach currently deemed popular and effective is using conditional generative adversarial network, also known shortly as cGAN [15]. It is adapted to perform image-to-image translation tasks with typically two networks: a generator and a discriminator[10]. The generator attempts to generate a *duplicated imitation* of the input data distribution from a noise distribution while the discriminator classifies whether the generator's input is fake, i.e. imitation from the generated distribution, or real, i.e. ground truth from the original distribution.

Previous research has focused on specific purpose for cross-image translation. Efros et al. [4] proposed a simple model to synthesize an image based on stitches of input images, which represents a traditional approach directly from input image. Fergus et al. [5] addressed specific problems of blurry image into higher crispness, which provided insights on retaining the crispness of duplicated imitation. Additionally, through user studies and an automated mechanism to select images, Chen et al. [3] developed a model to optimize image selection process before cross-image translation.

This project is based on an existing implementation of a network called Pix2Pix[10], a cGAN implementation based on U-Net architecture and convolution Markovian discriminator. The use of U-Net architecture [19], instead of the general encoder-decoder architecture is to improve the efficiency of processing input and output with higher resolution while facilitating the transfer of shared, low-level information directly across layers. The U-Net architecture enables cross-layer communication and therefore could more efficiently retain prominent input features, e.g. edge prominence.

In addition, Pix2Pix also employs a traditional approach, such as L1 distance, to imitate the ground truth and reduce blurring. In this case, L1 loss captures low-frequency crispness level [12]. The Markovian discriminator is employed to capture high frequency crispness level. It evaluates the image integrity with the unit of patch, namely the PatchGAN, with fewer parameters of image quality required while operating faster [13]. cGANs also combine the input data with the noise distribution to condition the resulting output to the original distribution. Pix2Pix applies this technique for image-to-image translation.

The advantage of this cooperated approach is that the Pix2Pix model can be applied to a varying types and depth of image-to-image translation problems without specifying loss functions specific to the application domain. Pix2Pix has already been applied to different image translation problems with notable success.

2. Problem statement

This project aims to evaluate the robustness of the Pix2Pix model by applying the Pix2Pix model to datasets consisting of cartoonized images. Using the Pix2Pix model, it should be possible to train the network to generate real-life images from the cartoonized images. The questions this project seeks to answer are:

1. Does the default set of hyper parameters of the original implementation work well on this cartoonized dataset?

2. Does the model properly recreate facial expressions and postures?

The project's experiment attempts to explore the robustness of the multi-purpose Pix2Pix model. The problem is to determine whether, under the standard hyper-parameters, Pix2Pix can accurately recreate the facial features of the images in the original distribution based on cartoonized version of the images.

2.1. Dataset

The dataset selected to train the Pix2Pix model is based on an adaption of the colorFERET [1] facial image data. The colorFERET database contains 11337 images of faces in different angles and postures. These images are taken from 1199 individuals and are typically used for facial recognition research.

The image data is provided in two versions: a higher resolution (512x768) and a lower resolution (256x384) version. In this project, the lower resolution version was used, in attempt to constrain the training duration.

2.1.1 Data Pre-Processing

This project utilizes a derivation of the colorFERET images. To obtain the actual data for training and testing the network, a modified version of an OpenCV cartoonizer script [2] was run on selected images to generate their cartoon versions. The pair of the cartoon and real images are then used as an input to the Pix2Pix model.

2.1.2 Data Filtering

After the first evaluation of the network, it was seen that the network seemed to performed badly due to the presence of some intermittent grayscale images, as well as images with low face-to-image ratio, where the person's face only cover a small portion of the full image. These images were later removed through manual inspection to create the filtered dataset. The filtered dataset contains a total of 8933 images which was divided into three parts: 60% for training, 20% for testing and 20% for validation.

2.1.3 Data Augmentation

For further evaluation through data augmentation, a set of images were used to create 4 different types of cartoons by varying the parameters of the OpenCV script such as image blurring, gray-scale percentage, etc. A total of 7,858 images were used for this purpose. In order to bring in domain adaptation and test its effect, the equivalent CycleGAN network was used for this purpose.

2.2. Expected Results

Motivated by the many successful applications [9] of Pix2Pix with different datasets, it is expected that the Pix2Pix model would perform similarly well, at least on a test data with similar distribution to the training data.

With data augmentation and domain adaptation, the network is expected to perform better for other test data types as well.

2.3. Evaluation

One of the challenges cited by the original Pix2Pix paper was a method of evaluating the model. In their examples, they used Amazon Mechanical Turks, a crowdsourcing marketplace, to evaluate if a person is able to distinguish whether an image is real or fake. For a more quantitative evaluation, they utilized the so-called FCN score, in which the authors trained a fully convolutional network to correctly label the original data with high confidence. This trained network was then used to classify the generated samples with the expectation that it would also accurately classify the generated data with a high confidence.

However, for this project, the result is initially analyzed visually and later we use the cosine dissimilarity to quantitatively evaluate our network.

3. Technical approach

This section details the use of cGANs and Pix2Pix, and the training of the network.

3.1. cGANs

Generative adversarial networks (GANs) [6] have been used to train generative models, which can generate data from some input. GANs use an adversarial model where two networks, namely the discriminator D and the generator G, play a sort of *minmax* game. The generator network attempts to generate synthetic data from a noise distribution and the discriminator estimates the probability that a data sample was generated from the original data distribution as opposed to the synthetic distribution. Equation 1 below captures the function which this adversarial model tries to optimize.

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

In Equation 1, the model aims to maximize the probability that the discriminator D assigns the correct label, whether the image is real or fake, to the input data while also minimizing the probability that data generated by the generator G is classified as real, as denoted by $\log 1 - D(G(z))$.

cGANs differ from the GAN model described above in that the adversarial model is conditioned using some additional data. This conditioning has the effect on exerting more control, i.e. providing a direction, on the data being generated. Equation details the optimization function for cGANs.

$$\min_G \max_D V(G,D) = \mathbb{E}_{x\sim p_{data}(x)[\log D(x\|y)]} + \mathbb{E}_{z\sim p_z(z)[\log 1-D(G(z\|y))]} \quad (2)$$

3.2. Pix2Pix

Pix2Pix [10] uses a cGAN model for image-to-image translation tasks of translating one possible representation of a scene to another. In the case of Pix2Pix, the generator G and discriminator D models are conditioned using the input image itself.

The original Pix2Pix implementation modifies the optimization function of cGANs from Equation 2 by adding an L1 loss term when training the generator G model. Equation 3 describes the optimization function for the generator G in Pix2Pix.

$$G^* = \arg\min_G \max_D L_{cGAN}(G, D) + \lambda L_{L1}(G) \quad (3)$$

where $L_{cGAN}(G, D) = \mathbb{E}_{z \sim p_z(z)[\log 1 - D(G(z\|y))]}$

and $\lambda L_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x, z)\|]$

The L1 loss in this case models the L1 per-pixel difference between the generated images and the ground truth. The authors picked L1 over L2 as the L2 distance produced more blurry results.

The authors also provided an implementation of the Pix2Pix model on GitHub [8]. This implementation was originally written in `lua`. However, this project uses a `PyTorch` [25] implementation of the paper.

In this implementation, a U-Net 256 network [18] was used as the generator network. For the discriminator, a PatchGAN network proposed by the Pix2Pix authors was used. This network restricts the evaluation scope to patches instead of the full image. In other words, it attempts to classify if each $N \times N$ patch in the image is real or fake. The authors also showed that N can be made much smaller than the image size without affecting the results.

3.3. Training the Model

When training the model, the input images were first scaled, cropped and then re-sized to 256x256 pixels. This is to condition the input for the U-Net 256 network which requires a resolution of 256x256 for the input images. Table 1 shows the default hyper-parameters that are used and Table 2 shows all the types of networks that were trained by variation of different hyper-parameters and datasets.

The training was carried out on the Google Cloud platform using an NVIDIA Tesla V100 GPU. It was performed for 200 epochs. The duration of training a Pix2Pix network for the default set of parameters (models 1, 2, 5 in Table 2) was approximately 8 hours, for a batch size of 8 (model 3) was 5 hours, for a batch size of 64 (model 4) was 2 hours, and training on a CycleGAN network for the default set of parameters (model 6) took about 10 hours.

Name	Value
Batch Size	1
Beta1	0.5
Beta2	0.9999
Discriminator Network	PatchGAN
Epsilon	1e-08
Generator Network	U-Net 256
Learning Rate	0.0002
L1 λ	100.0
Optimization Method	Adam
Weight Decay	0

Table 1: Training Configuration and Hyper-Parameters

No	Network	Dataset	Hyper-parameters
1	Pix2Pix	Original	No change
2	Pix2Pix	Filtered	No change
3	Pix2Pix	Filtered	Batch size = 8
4	Pix2Pix	Filtered	Batch size = 64
5	Pix2Pix	Augmented	No change
6	CycleGAN	Filtered	No change

Table 2: Variations used for training different models

4. Experiments and Results

This section describes the results of our experiment as we attempted to continuously improve the results by filtering the dataset, changing the hyper-parameters, and using a different network.

4.1. Initial Run

Using the default hyper-parameter settings shown in Table 1, the Pix2Pix model was trained using the cartoonized data. Upon visual inspection, many generated images resembled the original images. A sample of these can be seen in Figures 1a and 1b.

However, from visual inspection, the performance on the test data was not as good as the one observed during training. It was also seen that the network could not capture the facial expressions in grayscale images properly, resulting in a generated image that appeared blurry, as seen in Figure 1c. In addition, images with a low face-to-image ratio, in which the face did not occupy a large portion of the image also produced blurry results, as seen in Figure 1d.

These bad result cases of grayscale images and images with a low face-to-image ratio might be explained by the fact that the images not having enough data for an accurate translation. In the case of the grayscale images, only one channel is available and in the case of images with a low face-to-image size ratio, at the used resolution of 256x256, there simply is not enough data for a proper reconstruction.

(a) Sample result: input, generated, ground truth

(b) Sample result: input, generated, ground truth

(c) Grayscale image: input, generated, ground truth

(d) Low face-to-image ratio: input, generated, ground truth

Figure 1: Sample results from test on unfiltered dataset

(a) Sample Cartoon: Input, Generated, Ground Truth

(b) Sample Cartoon: Input, Generated, Ground Truth

(c) Sample Cartoon: Input, Generated, Ground Truth

Figure 2: Sample results from test on filtered dataset

4.2. Training with the Filtered Dataset

In a bid to obtain a better trained model, grayscale images and images with a low face-to-image ratio were filtered out. The Pix2Pix model was then retrained on this filtered dataset. Sample results from this run are shown in Figure 2. Visually, there was no apparent improvement over the run with the unfiltered dataset. This only removed the blurry results.

4.3. Varying the Batch Size

The previous experiments answered the question of how well the default set of hyper-parameters transfers to an application domain, different to those featured in the original paper. Due to the size of the dataset, the training time was quite large, 8 hours on the NVIDIA Tesla V100. To speed up the training time and see its effects on the quality of the results, the batch size was varied. The model was trained using a batch size of 8 and 64. The results from this run are shown in Figure 3 and Figure 4 respectively. It is seen that as the batch size increases, the visual quality of the output decreases. Figure 5 compares the training loss curves for the generator and discriminator with different batch sizes.

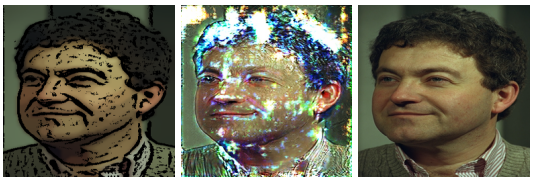
In Figures 5c and 5d, it can be seen that the generator GAN loss is both more stable and lower for higher batch sizes. The generator L1 loss is also more stable as the batch size increases. However, the inverse is the case for the discriminator. The discriminator loss for distinguishing between real and fake images is more unstable and on average higher as the batch size is increased. This could be interpreted to mean that at higher batch sizes, the discriminator is not as effective as it is with a batch size of 1. This would also translate to the generator appearing to perform better at higher batch sizes, since it is not being penalized well enough by its discriminator. This would explain why the visual quality of the generated images at the batch size of 1 is better than that generated by a higher batch size.

It was also observed that for a higher batch size, the L1 loss is higher than the GAN loss, a possible indicator that increasing the batch size should be accompanied with a change in other hyper-parameters.

4.4. Data Augmentation

The next question which needed to be answered was how well the model performs on out-of-distribution (OOD) samples. One technique to improve performance on such distributions involves constructing a common representative space [22] [14]. To achieve this, the script which was used to generate the synthetic data was slightly modified to generate different style of cartoons for the same image.

(a) Sample Cartoon: Input, Generated, Ground Truth



(b) Sample Cartoon: Input, Generated, Ground Truth

(c) Sample Cartoon: Input, Generated, Ground Truth

Figure 3: Sample results from test on filtered dataset with batch size 8

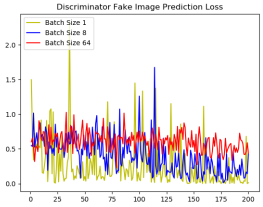
(a) Sample Cartoon: Input, Generated, Ground Truth

(b) Sample Cartoon: Input, Generated, Ground Truth

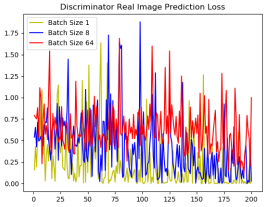
(c) Sample Cartoon: Input, Generated, Ground Truth

Figure 4: Sample results from test on filtered dataset with batch size 64

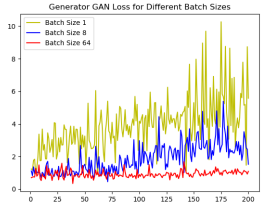
This allowed different samples to map to the same distribution. Figure 7 shows for the same image with different



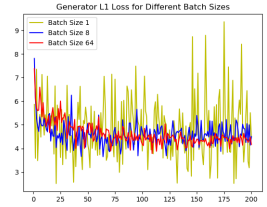
(a) Discriminator Loss (Fakes)



(b) Discriminator Loss (Real)



(c) Generator GAN Loss



(d) Generator L1 Loss

Figure 5: Loss graphs of the different batch sizes

(a) Input (b) Batch 1 (c) Batch 8 (d) Batch 64 (e) Gr. truth

Figure 6: Sample results on various batch sizes

cartoonization techniques and the corresponding generated image Figure 6 shows the performance of the model for different batch sizes, after training on this augmented dataset, on such an OOD sample.

The downside with this technique, however, is that the higher the difference between the OOD sample and the adversarial samples on which the network is trained the lower the quality of the generated image. Figure 8 shows the generated image when the input is a sketch of the original image.

4.5. Comparison to CycleGAN

The authors of Pix2Pix later attempted in improving the quality of image-to-image translation by building a state-of-the-art network called CycleGAN [24] that builds on Pix2Pix. The main difference with Pix2Pix is that CycleGAN learns the mapping without paired training examples. CycleGAN learns a mapping $G : X \rightarrow Y$ such that the distribution of images from $G(X)$ is highly identical to the distribution Y using an adversarial loss. As this mapping is highly under-constrained, it is coupled with an inverse mapping $F : Y \rightarrow X$ to introduce a cycle consistency loss such that $F(G(X)) \approx X$ and vice versa.

(a) Style 1: Cartoon, Generated, Ground Truth

(b) Style 2: Cartoon, Generated, Ground Truth

Figure 7: Training with different cartoonization techniques

(a) Sketch Input (b) Generated Image (c) Ground Truth

Figure 8: Image samples in case of hand drawn sketches (an OOD sample)

(a) Input (b) Pix2Pix (c) CycleGAN (d) Ground truth

Figure 9: Sample results on Pix2Pix vs CycleGAN

We trained our dataset on the CycleGAN framework and compared its test results, shown in Figure 9b, with those of Pix2Pix, Figure 9c. It can be observed here that CycleGAN outputs a much better image. CycleGAN does not remove the person's eyes and is able to approximate the color of the person's skin.

4.6. Quantitative Evaluation of Generated Images

Up to this point, the quality of the results has been reported only through a visual inspection. Quantitatively evaluating the generated results for GAN models is still an ongoing research area. The original paper utilized the so-called FCN Score. However, the network used to estimate the FCN score is especially suited for segmentation classification. In this report, two other evaluation techniques are utilized.

4.6.1 Cosine Dissimilarity

Cosine dissimilarity is a measure of the similarity between two vectors. Since the generated image and the target image are themselves representable as vectors, it is therefore possible to use the cosine dissimilarity to measure how close the two images are. Equation 4 gives a general formula for computing the cosine similarity between two vectors.

$$similarity = \cos\theta = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} \tag{4}$$

This measure does not penalize really small pixel differences, as would be the case in per-pixel comparison of the L1 distance. To account for negative values of $\cos\theta$, Equation 4 is modified slightly to become Equation 5

$$similarity = 1 - \cos\theta = 1 - \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} \tag{5}$$

In this new form, the values range from [0, 2] with a higher score indicating a higher level of dissimilarity. Figure 10 compares the similarity measure during training for the different batch sizes.

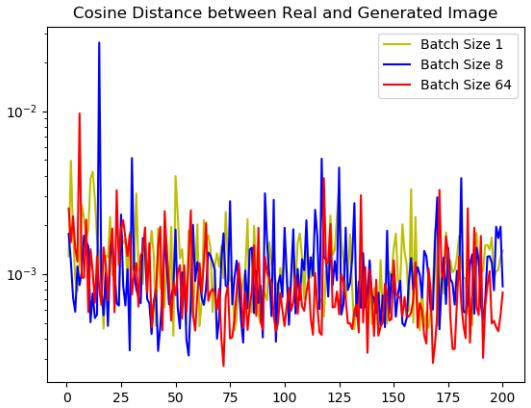


Figure 10: Cosine Similarity during training with different Batch Sizes

From Figure 10, it can be seen that on average, the cosine dissimilarity between the different batch sizes is the same. However, in line with the observations made through a visual inspection, Table 3 shows that the model trained with a batch size of 1 has the lowest cosine value and that

Batch Size	Mean Cosine Dissimilarity
1	0.00136
8	0.00622
64	0.01052

Table 3: Mean Cosine Dissimilarity for different Batch Sizes

the higher the batch size the worse the score. The disparity between this observation and that obtained when comparing the dissimilarity during training suggests that the higher batch size models do not generalize as well as the model with a batch size of 1.

4.6.2 Probability Score of an Age and Gender CNN

As an alternative to the FCN score, results were evaluated using a convolution neural network (CNN) trained to classify the gender of a facial image. A pre-trained Inception V3 model [20] was used as the classifier for this evaluation. For high quality generated images, the classifier should assign the correct label to the generated image with a similar level of confidence as it did to the original image. The gender label was used because the original FERET database was nearly evenly split between male and female. Other provided labels such as age and race did not provide as even a distribution.

The pre-trained inception model had an accuracy of 96% on the dataset used for training. Table 4 shows the mean absolute difference between predictions for the original images and the generated images for the test data for different batch sizes.

Results in Table 4 do not yield the same observation as the cosine similarity and the visual inspection evaluations, with the batch size 8 model having the least difference between the prediction probabilities for real and generated images. A possible explanation for this might be that gender classification is perhaps not the best label for evaluating the classification accuracy.

Batch Size	Mean Absolute Difference
1	0.075
8	0.060
64	0.078

Table 4: Mean absolute difference of image prediction probabilities for different batch sizes

5. Discussions

In this section, we discuss important aspects that greatly affected the results of our experiment.

5.1. Effect of batch sizes

Our finding that a higher batch size, while significantly reduces computation time, negatively affects the resulting generated image is reflected in existing literature. In practice, it has been seen that using a larger batch size results in a significant degradation in the quality of the model, as measured by its ability to generalize [11]. This lack of ability to generalize is mostly due to the fact that methods with high batch sizes tend to converge to sharp minimizers of the training function. Mishkin et al. [16] have also shown, in their studies on the performance of a CNN, that large mini-batch sizes results in a worse accuracy.

5.2. Downsides of Data Generation Techniques

The cartoons used in the training were unfortunately not real cartoons, but rather they were generated using OpenCV, which essentially added a filter to the image where the edges in the image are traced and the colors are simplified to resemble a cartoon picture. As a result, what the network learned was actually the features of this image filter and not the underlying features of image-to-image translation. This is evident when testing the network using an actual hand-drawn sketch, as seen in Figure 8b, where the trained Pix2Pix network outputs an image with the edges removed and some areas that are slightly blurred.

6. Conclusion

We have performed an experiment with several test rounds in order to evaluate the general robustness and applicability of Pix2Pix as solution to image-to-image translation tasks. On the test run where default parameters and configuration are retained, the result is less satisfying for images with non-amplified features, such as the lack of color or the lack of prominent facial features. This suggests that the model is susceptible to instability when the diversity of the image datasets varies.

In a later stage of the project, several alterations on data synthesis and hyper-parameters Were introduced. Image filtering assured a higher homogeneity within the distribution of the dataset, while the positive effect, in terms of computation time, of increasing the batch size is superseded by its damping effect on the discriminator. The performance of Pix2Pix on out-of-distribution samples also differs on the degree of variation between the original sample and the adversarial samples, which indicates unstable quality assurance under visual changes.

Moreover, a better evaluation methodology is required. The current methods employed can be easily categorized as subjective as the result and degree of variation from qualitative comparisons differs per individuals.

In general, the result of this report indicates acceptable robustness over varying image types and qualities. In case

where images are restricted with standard color and prominent facial features, the Pix2Pix model is a reliable image-to-image translation solution.

References

[1] Color feret database. https://www.nist.gov/itl/iad/image-group/color-feret-database. Accessed: 2019-05-27.

[2] Michael Beyeler. Cartoon effect using opencv. https://github.com/mbeyeler/opencv-python-blueprints/blob/master/chapter1/filters.py. Accessed: 2019-05-27.

[3] Tao Chen, Ming-Ming Cheng, Ping Tan, Ariel Shamir, and Shi-Min Hu. Sketch2photo: Internet image montage. *ACM Trans. Graph.*, 28(5):124:1–124:10, Dec. 2009.

[4] Alexei A. Efros and William T. Freeman. Image quilting for texture synthesis and transfer. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pages 341–346, New York, NY, USA, 2001. ACM.

[5] Rob Fergus, Barun Singh, Aaron Hertzmann, Sam T. Roweis, and William T. Freeman. Removing camera shake from a single photograph. *ACM Trans. Graph.*, 25(3):787–794, July 2006.

[6] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets, 2014.

[7] Aaron Hertzmann, Charles E Jacobs, Nuria Oliver, Brian Curless, and David H Salesin. Image analogies. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 327–340. ACM, 2001.

[8] Phillip Isola. pix2pix on github. https://github.com/phillipi/pix2pix. Accessed: 2019-05-27.

[9] Phillip Isola. Pix2pix project page. https://phillipi.github.io/pix2pix/. Accessed: 2019-05-27.

[10] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks, 2018.

[11] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *CoRR*, abs/1609.04836, 2016.

[12] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. *CoRR*, abs/1512.09300, 2015.

[13] Chuan Li and Michael Wand. Precomputed real-time texture synthesis with markovian generative adversarial networks. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 702–716, Cham, 2016. Springer International Publishing.

[14] Wulfmeier Markus, Bewley Alex, and Posner Ingmar. Addressing appearance change in outdoor robotics with adversarial domain adaptation.

[15] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets, 2014.

[16] Dmytro Mishkin, Nikolay Sergievskiy, and Jiri Matas. Systematic evaluation of CNN advances on the imagenet. *CoRR*, abs/1606.02228, 2016.

[17] Xinlei Pan, Yurong You, Ziyan Wang, and Cewu Lu. Virtual to real reinforcement learning for autonomous driving. *arXiv preprint arXiv:1704.03952*, 2017.

[18] Olaf Ronneberger, Philipp Fischer, and Thomas Brox Alexei. U-net: Convolutional networks for biomedical image segmentation, 2015.

[19] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing.

[20] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015.

[21] Yi-Hsuan Tsai, Kihyuk Sohn, Samuel Schulter, and Manmohan Chandraker. Domain adaptation for structured output via discriminative representations. *arXiv preprint arXiv:1901.05427*, 2019.

[22] Ganin Yaroslav, Ustinova Evgeniya, Ajakan Hana, Germain Pascal, Larochelle Hugo, Laviolette Francois, Marchand Mario, and Lempitsky Victor. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17, 2016.

[23] Ziqiang Zheng, Chao Wang, Zhibin Yu, Nan Wang, Haiyong Zheng, and Bing Zheng. Unpaired photo-to-caricature translation on faces in the wild. *Neurocomputing*, 355:71–81, 2019.

[24] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *CoRR*, abs/1703.10593, 2017.

[25] Jun-Yan Zhu. Pytorch pix2pix on github. https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix. Accessed: 2019-05-27.