DeepDraw: Fast Domain Translation from Sketches to Photo-Realistic Faces

Yingsi Qin
Columbia University
New York, NY
yq2285@columbia.edu

Abstract

Translating from contour sketches to photo-realistic faces is an image-to-image translation problem, which has gained a lot of popularity using conditional generative adversarial networks. However, image-to-image paired supervised learning requires expensive gathering of human-drawn sketches and as a result, previous works addressing this problem have been using edges automatically generated from traditional edge detection algorithms that detect high-frequency information only.

We instead propose to infer contour sketches from real images using a pre-trained GAN-based network [9]. Moreover, instead of training on sketch-image pairs, we train on the two domains without pairing specific images. First, we plan to first transfer-learn a pre-trained Cycle-Consistent Adversarial Network [18] on our generated sketch dataset and a real-face image dataset. Then, we develop a web application as the user interface to allow users to draw and demonstrate live generation of face images from sketches.

Our method has several advantages: (1) using inferred contour sketches instead of traditionally generated edges allow us to train on human-like contour drawings tat account for image contextual understanding, (2) unpaired image-to-image translation allows the network to infer both the common latent features between domains and the unique features to each domain, (3) the cycle-consistency ensures that the output face is invertible. We cross compared our results to previous related works via quantitative and qualitative evaluation methods (see section 5).

1. Motivation

Outlining the contours of a face is usually the first step in drawing a face. Not only in art, in real life, we do not always have a photo of a person, and therefore can only infer the real photo from the contour sketches we can make. Moreover, the most satisfactory outcome usually comes after multiple feedback rounds, which have been a very slow process. In an attempt to facilitate such visual retrieval process, we present this study to experiment with ways to improve the output fidelity while reducing the huge computational cost required by most GANs.

2. Related Works

Conditional Generative Adversarial Networks (cGANs). A cGAN is a GAN except that it conditions on a specific set of input information, analogous to how variational autoencoders are used to condition on the inputs. It also has a generator-discriminator pair with the generator taking an image from the source domain and outputting a fake image aiming to fool the discriminator that it belongs to the target domain. It is trained on an adversarial loss. Various applications include image inpainting using context encoders [11], creating super resolution images [1], face de-aging [17], text-to-image translation [3], clothing translation [16], video prediction [13], 3D Object Generation [14], etc. The primary CycleGAN [18] that we adopt belongs to this broader category of cGAN.

Sketch-to-Image Translation. The major increase in the number of studies researching sketch-to-image translation happens after Isola et al. published their pix2pix network [7]. pix2pix uses a conditional generative adversarial network to learn a one-to-one mapping from input to output images that belong to different domains. They showed that pix2pix can be applicable in a lot of applications and sketch-to-image was one of them. Since then, numeral studies experimented with tweaking around the architectures and training pipelines to obtain higher-fidelity outputs, such as Scribbler [12] and SketchyGAN [2]. Among them, Contextual GAN is one that outputs photo-realistic results [10]. It uses a joint image completion approach to allow more freedom in the generated features so that they do not strictly align with the input features as previous conditional GANs do. Moreover, Ghosh et al. [4] made possible interactive sketch & fill by using a GAN-based residual

Encoder-Decoder model that adapts from MUNIT [6] and constructing a gating-based approach for class conditioning and shape completion. Their model consists of a single generator network. Despite that numerous efforts have been made in this field, to our observation, all of the existing methods (1) use supervised learning on sketch-to-image pairs and (2) use edges generated using traditional edge detection algorithms.

Unpaired Image-to-Image Translation. Zhu et al. showed that we do not have to train on paired images. Instead, they showed that training on domains work even better [18]. There are two highlights of this unpaired translation method: (1) they allow the generators and discriminator to learn features common and unique to both domains, and (2) they use a cycle-consistency loss in addition to the adverial loss to ensure that the generated image is invertible. We used this method in our project.

3. The Limits: Datasets Generation Pipeline

3.1. Face Dataset

We used the open-source CelebA dataset [15]. This dataset contains 202,599 number of face images of various celebrities.

Added in re-submission: The limit we impose is that we use this dataset by starting with 2k image samples for training and 500 images for testing, and then experimentally increase the size of the data used if needed. The amount of data that we eventually used depend on our empirical assessment of (1) the model performance, (2) the training time, and (3) if increasing the dataset increases the performance.

The faces in the dataset are well centered. Although there are differences in the backgrounds of each images, we did not filter this dataset as we do not want filtering to reduce the variance of our distribution. However, we normalized the images, downsize the images if necessary for faster runtime, and augment the dataset by random flipping and croppping to increase the variance of the distribution and make it more robust to test cases. We downloaded the dataset from Kaggle[1].

3.2. Sketch Dataset

Given the popularity in using cGANs to generate realistic-looking images from sketches, as far as we have seen, none of the existing cGAN methods (1) uses a well-trained network to generate the sketch dataset as they all used automatic edge detection methods such as Canny which generates not-human-drawn-looking edges, and (2) train a generative neural network using the human-drawn-looking edge photos.

[1]https://www.kaggle.com/jessicali9530/celeba-dataset

Therefore, we used the contour generator network[2] by Li et al. [9], who have shown that the existing cGAN methods do not work on sketch generation out-of-the-box, to generate human-drawn-like contour sketches from the images in the CelebA dataset. Shown in Figure 1 is the output of this contour generator network compared to that using the Holistically-nested edge detection (HED) method. We can see that HED only captures high frequency signals without understanding the image, while the contour generator outputs salient inner boundaries while reflecting the imperfections in ground truth human drawing.

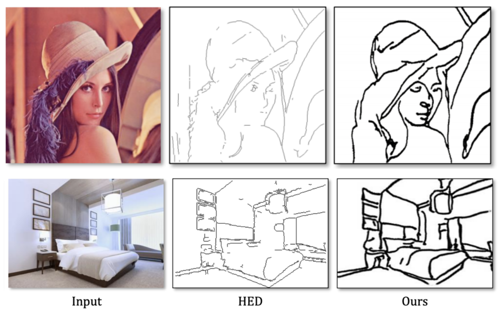


Figure 1. Contours generated examples taken from [9]. Unlike traditional edge detectors which only capture high frequency signals, the contour generation method we adopt can (1) generate the most salient inner boundaries, (2) occluding contours, and (3) reflect imperfections in a human-drawn contour sketch. For example, we can see that the ceiling is not a perfect straight line.

3.3. Background Masking

Instead of using the CelebA dataset [15] which contains 202,599 number of face images of various celebrities, we found a better alternative, which was to impose a mask that exclude out the background information in the CelebA dataset. We found that the CelebAMask-HQ [8] is such a dataset that contains 30,000 high-resolution face images selected from the CelebA dataset by following CelebA-HQ. Each image has segmentation mask of facial attributes corresponding to CelebA. We show some face and mask pair examples in Figure 4.

The masks of CelebAMask-HQ were manually-annotated with the size of 512 x 512 and 19 classes including all facial components and accessories such as skin, nose, eyes, eyebrows, ears, mouth, lip, hair, hat, eyeglass, earring, necklace, neck, and cloth. The dataset contains individual components of the face for each face image. In figure we show how the face is segmented into different components.

[2]https://github.com/mtli/PhotoSketch

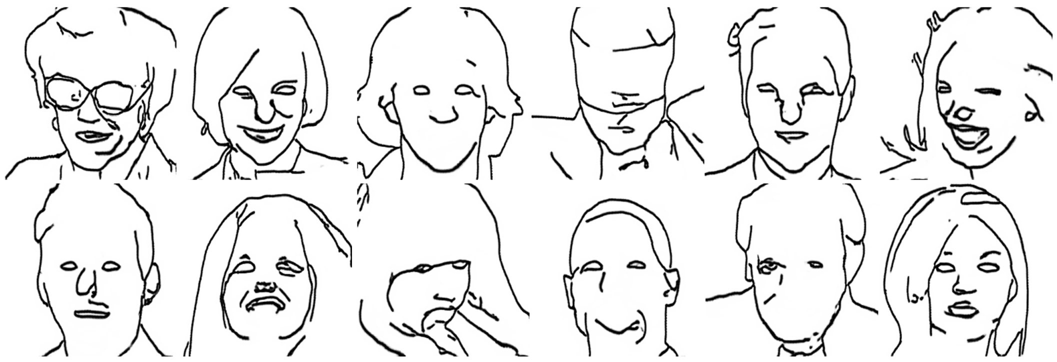


Figure 2. Visualization of our generated sketch dataset

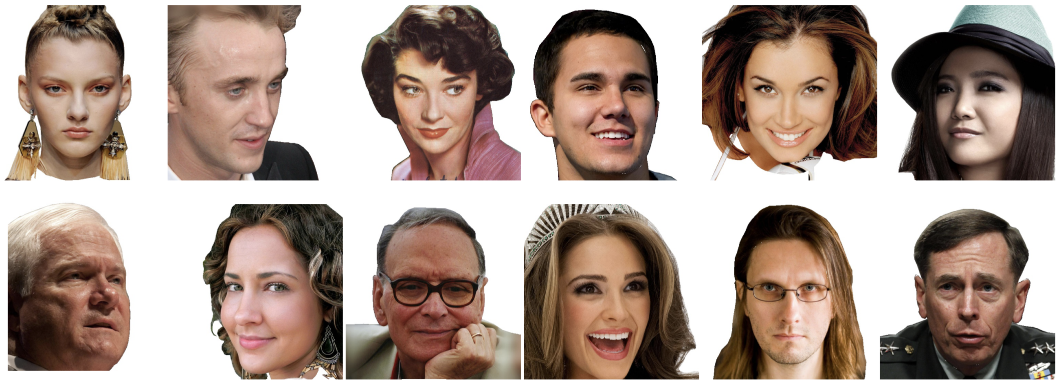


Figure 3. Visualization of our masked face dataset

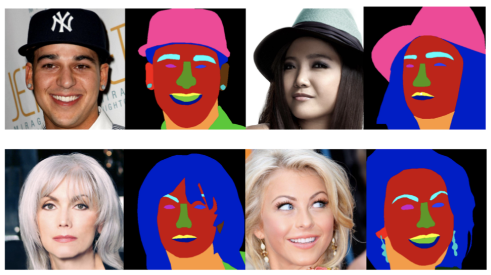


Figure 4. Face and mask pair examples from [8].

3.4. Complexity and Diversity of Our Data

The complete data generation flow. We incorporate the aggregated mask of all facial components in our pipeline by multiplying it as an alpha mask for the face images so that all irrelevant background information is filtered out. More details are discussed in Section 4. As mentioned in Section 3.2, we then used the contour generator network by Li et al. [9] to generate human-drawn-like contour images for 2k image samples from the CelebAMask-HQ dataset. The two output datasets of this pipeline is shown in Figure 2 and Figure 3.

Complexity and diversity. Unlike a lot of the existing studies that uses the CelebA dataset, we used the mask datatset to deliberately allow a great variety of facial features, including but not limited to: earrings of different sizes, different hair decorations, clowns, glasses, hats and caps, different look angles, neckalces, etc. This greatly increased the diversity in our data. The complexity comes in when the sketches are generated. From the example shown above, it is clear that a lot of the sketches are not even a complete drawing. This is because we want to allow a greater tolerance of the network being able to reconstruct an instance of the other domain from one domain - to increase the network robustness.

4. Methods and Anticipated Results

4.1. Model Architecture

Previous methods have mostly used one-to-one paired supervised training for a sketch-and-image pair. However, it can be difficult and expensive to obtain paired data, and sketches often times can consist of different styles of drawing. Therefore, we adopt the unpaired image-to-image translation method [18] which translates between domains instead of paired images, as shown in Figure 6. Just like how humans do not necessarily need to have seen the one-to-one edge-to-image pairs in order to imagine a realistic photo from sketches, the Cycle-Consistent Adversarial Network (CycleGAN) [18] learns the translation between the sketch domain and the face photo domain.

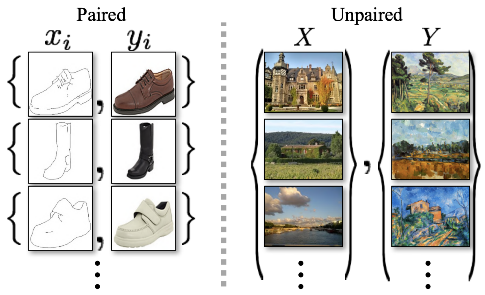


Figure 6. Paired vs. Unpaired Training. CycleGAN learns between domain X and Y, instead of image pairs (x_1, y_1). Image Taken from [18].

The goal of CycleGAN is to learn mapping functions between domain X and Y which consist of training samples $x_{i\,i=1}^{\,N}$ with $x_i \in X$ and $y_{i\,i=1}^{\,M}$ with $y_i \in Y$. The mapping functions are $G : X \rightarrow Y$ and $F : Y \rightarrow X$, as indicated in Figure 5. Let D_X and D_Y be the two adversarial discriminators, then D_X aims to discriminate between images x and the generated images $F(y)$. Similarly, D_Y discriminate between images y and the generated images $G(x)$. We want the generated image to be invertible so that $G(F(x)) = x$. We pictorially show this in Figure 7.

4.2. Objective Function

Let $x \sim p_{data}(x)$ and $y \sim p_{data}(y)$ denote the data distribution and the rest of the notations follow Figure 5. Then, the full objective function includes two kinds of losses: adversarial loss and cycle-consistency loss. The impact of cycle-consistency loss can be observed in Figure 7, that $F(G(x)) \approx x$. The cycle-consistency loss is:

$$\begin{aligned} L_{cyc}(G, F) = & \underset{x \sim p_{data}(x)}{\mathbb{E}} [||F(G(x)) - x||_1] \\ & + \underset{y \sim p_{data}(y)}{\mathbb{E}} [|G(F(x)) - y|_1] \end{aligned} \tag{1}$$

The adversarial loss is:

$$\begin{aligned} L_{GAN}(G, D_Y, X, Y) = & \underset{y \sim p_{data}(y)}{\mathbb{E}} [log D_Y(y)] \\ & + \underset{x \sim p_{data}(y)}{\mathbb{E}} [log(1 - D_Y(G(x)))] \end{aligned} \tag{2}$$

Then, the full objective function is:

$$\begin{aligned} L(G, F, D_X, D_Y) = & L_{GAN}(G, D_Y, X, Y) \\ & + L_{GAN}(F, D_X, X, Y) \\ & + \lambda L_{cyc}(G, F) \end{aligned} \tag{3}$$

where λ controls the relative importance of the two objectives. We aim to solve for the optimal G and F which give the argmin of max $L(G, F, D_X, D_Y)$. During our training, we take the pre-trained network from their code base[3], and transfer-learn it on our sketch and face datasets. We started with their default training hyper-parameters and then investigate changes if necessary.

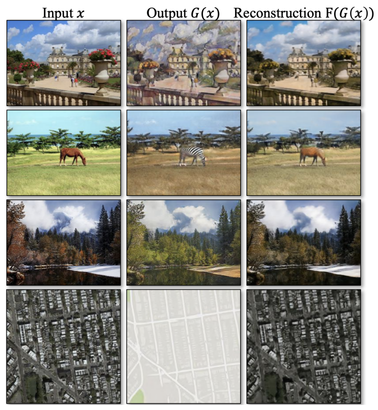


Figure 7. Pictorial examples of $F(G(x)) \approx x$. Image Taken from [18].

4.3. Results, effectiveness, and goodness since the Prototype

While we present the evaluation of the results of our compressed and uncompressed CycleGAN in the next section, it is worth mentioning that for the CycleGAN, we expect to see generated faces to be more photo-realistic as compared to previous methods. We also expect that the generated face should be invertible to the contour sketch. For the compressed CycleGAN, we expect the results from the compressed CycleGAN to be similar to that from the uncompressed.

[3]https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix

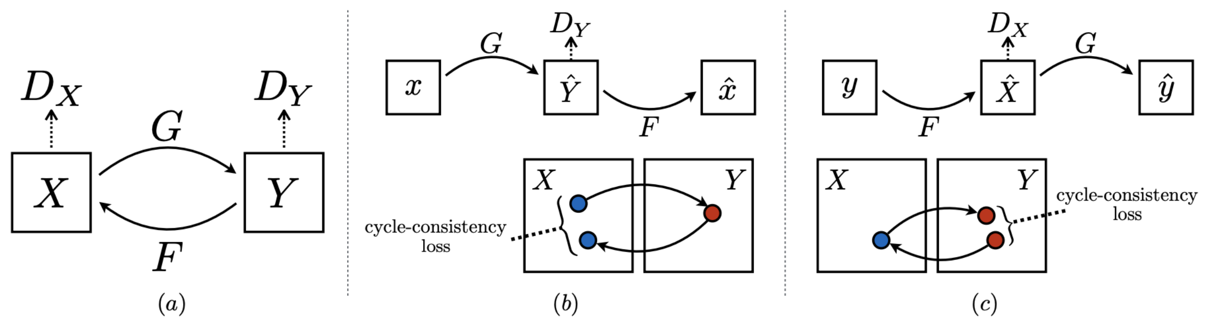


Figure 5. Illustration of cycle-consistency to ensure the generated image is invertible to the input image. (a) we want $G(F(x)) = x$; (b) forward cycle-consistency loss of $F(G(x)) \approx x$; (c) backward cycle-consistency loss of $G(F(y)) \approx y$. Image Taken from [18].

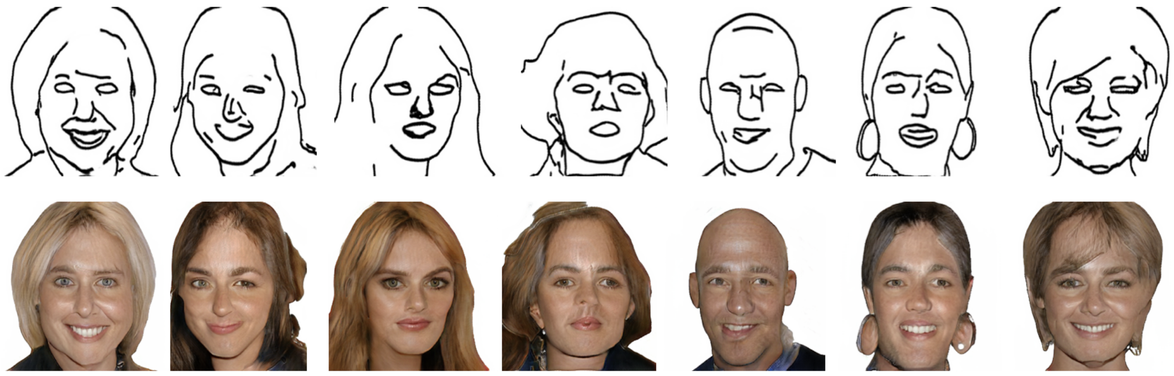


Figure 8. Testing results of our network. Top row: Face Sketch Input. Bottom row: Generated Face Output. Note that we only use the one generator that generates from the sketch domain to the face image domain.

4.3.1 The Training Challenges

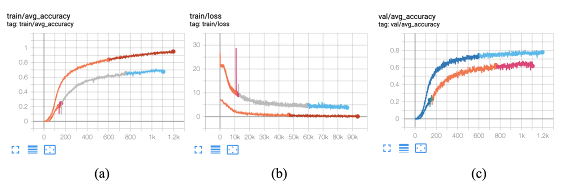


Figure 9. A screenshot of the training progress on Tensorboard.

The entire pipeline is constructed on Google Cloud Platform in Tensorflow 2.0 with an NVIDIA tesla T4 GPU. A test pass take 4.957569774 seconds to finish. Training progress of multiple runs with different parameters turned is shown in Figure 9. The curves of different colors indicate different runs started at different time. We can see that for the blue run, the validation accuracy is around 0.8. We are still in the process of improving this accuracy. An illustration of the test pass is shown in Figure **??**. Details are discussed in the demo.

Since the prototype, we were able to compress the runtime down from 4.17 seconds to 2.1 seconds for real-time interaction. We decided that compression is unnecessary as the entire network is done improved.Additionally, we increased the complexity and diversity of the input dataset since the prototype generation. The network is more robust and effective towards the kinds of user input it takes. We discuss more of this in detail in the evaluation section.

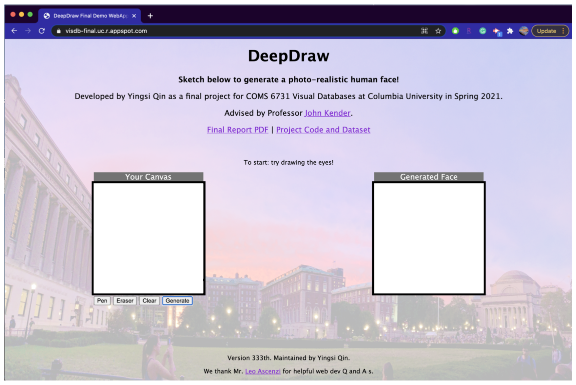
We show in Figure 8seven random examples of the testing outputs using sketches that the network had neven seen. Note that we only use the one generator that generates from the sketch domain to the face image domain.

4.3.2 The Web Application User Interface

Our user interface is an interactive real-time web application where the user can sketch a face and generate a photorealistic face image output. The Web Application is shown in Figure 10. Details are discussed in the demo. It is available here: `https://visdb-final.uc.r.appspot.com/`

Time and effort divested in this. It is worth mentioning that the Web Application development of the demo, from start to finish, took in total two weeks, about 80 hours.

The author had no experience in web development before, but in order to evaluate the performance of our network and have a place to demonstrate the goodness and effectivess, we decided that it is worth spending a lot of time and effort into this.



DeepDraw

Sketch below to generate a photo-realistic human face!

Developed by Yingsi Qin as a final project for COMS 6731 Visual Databases at Columbia University in Spring 2021.

Advised by Professor John Kender.

Final Report PDF | Project Code and Dataset

To start: try drawing the eyes!

Version 333th. Maintained by Yingsi Qin.

We thank Mr. Leo Ascenzi for helpful web dev Q and A s.

Figure 10. Our DEMO Web Application User Interface

5. Evaluation

5.1. Quantitative Measure: FID score

We used the Frechet Inception Distance (FID) score [5] as a quantitative measure to evaluate how similar our generated face photos are as compared to the ground truth face photos. This score is defined as d^2:

$$d^2 = ||mu_1 – mu_2||^2 + Tr(C_1 + C_2 – 2 * sqrt(C_1 * C_2))$$

where mu_1 and mu_2 are the feature-wise mean of the real and generated images, C_1 and C_2 are the covariance matrix for the real and generated feature vectors, often referred to as sigma, $||mu_1 – mu_2||^2$ are the sum squared difference between the two mean vectors, and Tr refers to the trace of a matrix.

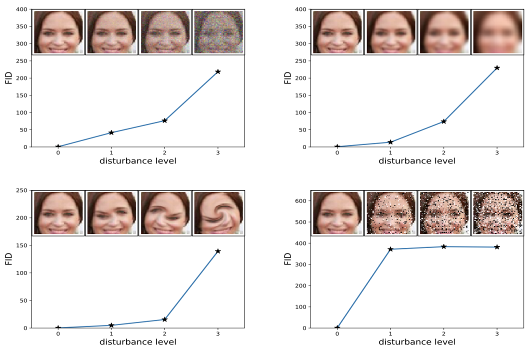


Figure 11. FID scores vs different distortion types and levels. Image Taken from [5].

Figure 11 shows FID under different distortion types: Gaussian noise (upper left), Gaussian blur (upper right), swirled images (lower right), and salt and pepper noise (lower right). We can tell that with stronger distortion the FID score monotonically increases.

5.2. Qualitative Measure: Human Perceptual Study using A/B Testing

Participant Selection. We conducted a small human perceptual study of 7 participants, all of whom indicated in their initial survey of some level of artistic mastery and aesthetic appreciation. We wanted to make sure that participants can draw face contours so to evaluate the effectiveness of our network.

5.2.1 Phase 1

Evaluation on which image looks the most realistic. Metrics shown in Table 1.

Method	%Ranked Realistic±Error
Real Image	93%±2.79%
Ours	7%±1.33%

Table 1. Phase 1 statistics from feedback of 7 participants.

5.2.2 Phase 2

Evaluation on which image (1) fits your drawing the most and (2) is the most realistic. It is worth mentioning that this will be kept for future work.

Method	%Realistic±Error	%Fits±Error
pix2pix	%±	%±
CycleGAN+Canny	%±	%±
Ours	%±	%±

Table 2. Phase 2 participant feedback. Placeholder Table.

Overall, the test users demonstrated a preference towards the real images that they appeared to be more realistic. However, their level of satisfaction indicated of our demo was above what they expected. In our analysis, we discovered that the differences between the user study output and the test output was caused by the different stroke sizes used in training and at demo time. The smallest storke size that web application UI allows is 1 px and with a 1px smoothing effect applied to it, it appears like 3px wide, which is wider than those in the training data. Thus the network is not trained to be generalizing well for this type of thick-stroke sketches.

6. A Future System and Reflection

As discussed previously, a future system would include (1) more diversity and complexity in the contour dataset, so that it includes a wide variety of drawing styles and edge styles, (2) more complexity in the selected facial features but less breath diversity in the features selected given a small dataset, or (3) increase dataset training size to even bigger and more computational resources, (4) potentially compress the network to under 100ms per run, (5) include an additional phase in the evaluation process so that comparison can be drawn between this work and the existing State-of-the-Art, and more over, (6) increase the size of the input canvas on the demo so that the relative stroke size can be the same as those in the training data.

A lot was learned for this final project. The author had never trained a GAN before, but now the author knows how to train a GAN, dynamically, and use Google Cloud Storage as well for effective training. Moreover, the author learned how to conduct effective user studies. The authorgot to agilely develop her UI during the user studies by taking in user feedback. Besides, the author learned the idea of "visual retrieval" and the generative neural network equivalence of it. The author also learned that training of a neural network should start small, from a trial dataset, then scaled up to a training dataset, in order to detect any early malfunctions in the pipeline.

7. Acknowledgement

Literature references are cited in the References section. The model was trained on Google Cloud Platform VMs and coupons from the class may be requested if needed. Usage of datasets were discussed in section 2. The code and the generated databases are available in our storage bucket on Google Cloud Storage here[4].

This is a project progress report for the course COMS 6731. This contributing author of this prototype is the only student working on this project, including but not limited to, literature review, acquisition and processing of the CelebA dataset, generating the sketch dataset, designing the methods, training/testing of the network, developing the prototype, conducting the human perceptual study etc. Throughout the process, Professor Kender supervises this research effort and provides advisory help.

References

[1] Huang Bin, Chen Weihai, Wu Xingming, and Lin Chun-Liang. High-quality face image sr using conditional generative adversarial networks, 2017. 1

[2] Wengling Chen and James Hays. Sketchygan: Towards diverse and realistic sketch to image synthesis, 2018. 1

[3] Ayushman Dash, John Cristian Borges Gamboa, Sheraz Ahmed, Marcus Liwicki, and Muhammad Zeshan Afzal. Tac-gan - text conditioned auxiliary classifier generative adversarial network, 2017. 1

[4] Arnab Ghosh, Richard Zhang, Puneet K. Dokania, Oliver Wang, Alexei A. Efros, Philip H. S. Torr, and Eli Shechtman. Interactive sketch fill: Multiclass sketch-to-image translation, 2019. 1

[5] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium, 2018. 6

[6] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation, 2018. 2

[7] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks, 2018. 1

[8] Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. Maskgan: Towards diverse and interactive facial image manipulation, 2020. 2, 3

[9] Mengtian Li, Zhe Lin, Radomir Mech, Ersin Yumer, and Deva Ramanan. Photo-sketching: Inferring contour drawings from images, 2019. 1, 2, 3

[10] Yongyi Lu, Shangzhe Wu, Yu-Wing Tai, and Chi-Keung Tang. Image generation from sketch constraint using contextual gan, 2018. 1

[11] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. Context encoders: Feature learning by inpainting, 2016. 1

[12] Patsorn Sangkloy, Jingwan Lu, Chen Fang, Fisher Yu, and James Hays. Scribbler: Controlling deep image synthesis with sketch and color, 2016. 1

[13] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating videos with scene dynamics, 2016. 1

[14] Jiajun Wu, Chengkai Zhang, Tianfan Xue, William T. Freeman, and Joshua B. Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling, 2017. 1

[15] Shuo Yang, Ping Luo, Chen Change Loy, and Xiaoou Tang. From facial parts responses to face detection: A deep learning approach, 2015. 2

[16] Donggeun Yoo, Namil Kim, Sunggyun Park, Anthony S. Paek, and In So Kweon. Pixel-level domain transfer, 2016. 1

[17] Zhifei Zhang, Yang Song, and Hairong Qi. Age progression/regression by conditional adversarial autoencoder, 2017. 1

[18] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks, 2020. 1, 2, 3, 4, 5

[4]https://console.cloud.google.com/storage/browser/visdb-final-dataset