SKETCH TO FACE TRANSFORMATION FOR CRIMINAL INVESTIGATION

Submitted in partial fulfillment of the requirements of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER ENGINEERING

By

Mr. Aryan Nayak

Ms. Rachaell Nihalaani

Mr. Sparsh Nagpal

Guide:

PROF. VAIBHAV AMBHIRE

(Assistant Professor, Department of Computer Engineering, TSEC)

Computer Engineering Department

Thadomal Shahani Engineering College

University of Mumbai

2021-2022

Abstract

Criminal investigations often have sketches as the only reference to identify a criminal suspect. Sketch images contain basic face profile information but lack detailing, as is contained in photogenic images. Thus, recognizing actual human faces from those sketches becomes difficult, and it would help to generate a face image from these sketches.

The implementation of computer vision into this will reduce the human effort to relate a black and white drawing to actual faces by means of image translation. Previous works have handled the task of transforming viewed sketches into mugshot images, and focused on forensic sketches. In this study, we aim to transform the sketches into realistic facial photographs.

With the input sketch the output of common image-to-image translation follows the input edges due to the hard condition imposed by the translation process. Instead, we propose to use sketch as weak constraint, where the output edges do not necessarily follow the input edges. We address this problem using a novel joint image completion approach, where the sketch provides the image context for completing, or generating the output image.

We have created a model using contextual Generative Adversarial Networks (GANs). It takes an artistic sketch of a person's face as the input image, enhances certain features of it and transforms it into a realistic photograph of the face of that person. We have used the CUHK Face Sketch Dataset to train our model.

Chapter 1

Introduction

1.1 Introduction

Facial composite sketches are done by the forensic sketch artists to help law enforcement officers to identify criminal suspects. Sometimes the sketches alone fail to provide detailing, therefore it's essential to match these photos for preciseness. Criminal Investigation has recently advanced its technical capabilities and methodologies. Criminal identification using face sketches is still an implementation that requires good human visual and memory skills. Unless we are able to generate images from thoughts efficiently, we are dependent on the memory and description given to the person ahead creating the sketch. Sketch images contain basic face profile information but lack detailing, as is contained in photogenic images. Thus, recognizing actual human faces from those sketches becomes difficult.

The implementation of computer vision into this will reduce the human effort to relate a black and white drawing to actual faces by means of image translation. Previous works have handled the task of transforming viewed sketches into mugshot images, and focused on forensic sketches. In this study, we aim to transform the viewed sketches into realistic facial photos.

In recent years, a wide variety of image transformation tasks have been solved by training deep convolutional neural networks (CNNs). The networks receive an input image and transform it into a corresponding output image; both detailed information and complex textures of the images are required in this process. Several researchers have recently tackled these individual tasks with specific mechanisms.

1.2 Aims and Objectives

Our objective is to create a model that takes an artistic face-sketch of a person as the input image, enhances certain features of it and transforms it into a realistic (or as close as possible) photograph of the face of that person. Train the dataset over a sizable dataset for better results.

Our model will utilize the Generative Adversarial Networks (GANs) which will utilize the generator classifier model and a dataset of face images and their respective sketches to first build an efficient classifier model and then additional noise input to make a learning generator which on multiple epochs will accurately recreate the facial image from the given input sketch.

1.3 Scope

Common approaches used now in conditional generative adversarial networks (cGAN) incorporate hard conditions like pixel-wise correspondence alongside the translation process, which makes the output strictly align with the input edges. This can be highly problematic in sketch-to-image generation when the input is a free-hand sketch.

Our goal of sketch-to-image generation is to automatically generate a photographic image of the hand-sketched object. Even a poorly drawn sketch allows non-artists to easily specify an object's attributes in many situations which may be clumsy to specify in verbose text description. On the other hand, the translation should respect the sparse input content, but might need some deviation in shape to generate a realistic image. In order to tackle these challenges, we propose a novel contextual generative adversarial network for sketch-to-image generation. We pose the image generation problem as an image completion problem, with sketch providing a weak contextual constraint.

In conventional image completion, the corrupted part of an input image is completed using surrounding image content as context. A generative adversarial network is trained to learn the joint distribution and capture the inherent correspondence between a sketch and its corresponding image using the defined joint image. This approach encodes the "corrupted" joint image into the closest "uncorrupted" joint image in the latent space via back propagation, which can be used to predict and hence generate the output image part of the joint image. To infer a closest mapping, we use sketch as a weak constraint and define our objective function which is composed of a contextual loss as well as traditional GAN loss. We also propose a straight-forward scheme for better initialization of sketches.

This novel approach has several advantages. There are no separate domains for image and sketch learning; only one network is used to understand a joint sketch-image pair which is a single image. This is in stark contrast with image translation where only sketch is treated as input. By using a weak sketch constraint, while related to its input edges, the generated image may exhibit different poses and shapes beyond the input sketches which may not strictly correspond to photographic objects. From the joint image's point of view, image and sketch are of no difference, so they can be swapped to provide the context for completion for the other. Thus, exactly the same sketch-to-image generation approach/network can be used for the reverse or image-to-sketch generation

Our framework is generic which can employ any state-of-the-art generative model. Capitalizing on the GAN for image completion, we have a two-phase recipe to learn the sketch-image correspondence inherent in a joint image as well as imposing the weak sketch constraint. For training, the network learns the sketch-image correspondence using uncropped joint images.

In completion, we search for an encoding of the provided corrupted image using only the sketch to provide the weak context for "completing" the image based on a modified objective. This encoding is then used to reconstruct the image by feeding it to the generator which generates the photographic object from the sketch.

Chapter 2

Review of Literature

2.1 Domain Explanation

2.1.1 Adversarial Learning

We have experimentally validated that deep learning models are highly vulnerable to attacks that are based on small modifications of the input to the model at test time. Suppose you have a trained classifier that correctly recognizes an object in an image with the correct label.

It's possible to construct an adversarial example, which is a visually indistinguishable image. These adversarial images can be constructed by noise perturbation. However,

the image is classified incorrectly. To address this problem, a common approach is to inject adversarial examples into the training set (adversarial training) [1]. Hence, the neural network's robustness can be increased. This type of example can be generated by adding noise, by applying data augmentation techniques or by perturbing the image in the opposite direction of the gradient (to maximize loss instead of minimizing it). Nevertheless, these types of techniques are a little bit hand-crafted so there will always be a different perturbation that can be applied to fool our classifier. Now let's think the other way around!

What if we are not interested in having a robust classifier, but rather we are interested in replacing the hand-crafted procedure of adversarial examples? In this way, we can let the network create different examples that are visually appealing. So, that's exactly where the generative term comes into play. We focus on producing representative examples of the dataset instead of making the network more robust to perturbations! Let us use another network to do this work for us.

2.1.2 Vanilla GAN

Since a generator is simply a neural network, we need to provide it with an input and decide what the output will be. In the simplest form, the input is random noise that is sampled from a distribution in a small range of real numbers. This is what we call latent or continuous space. However, since I use the word random it means that every time I sample from a random distribution, I will get a different vector sample. That's what stochasticity is all about. Such input will provide us with a non-deterministic output. As a consequence, there is no limit to the number of samples we can generate! Stochasticity is also the reason why you see the symbol of expected value in the papers related to Generative Adversarial Learning (GAN). The output will be a generated image for image generation or something that we would like to generate. The main difference is that now we focus on generating representative examples of a specific distribution (i.e. dogs, paintings, street pictures, airplanes, etc.)

The discriminator is nothing more than the classifier. However, instead of classifying an image in the correct class, we focus on learning the distribution of the class (let's say producing building images). Therefore, since the desired class is known, we want the classifier to quantify how representative the class is to the real class distribution. That's why discriminators output a single probability, wherein 0 corresponds to the fake generated images and 1 to the real samples from our distribution.

An ideal generator would produce indistinguishable examples which means an output of 0.5. In terms of game theory, this adversary is called a 2-player minmax game. In game theory, a min-max game or Nash equilibrium is a mathematical representation of a non-cooperative game involving two players, in which each player is assumed to know the equilibrium strategies of the other player, and no player has anything to gain by changing only their own strategy.

Generator G attempts to produce fake examples that are close to the real distribution so as to fool Discriminator D, while D tries to decide the origin of the distribution. The core idea that rules GANs is based on the "indirect" training through D that is also getting updated dynamically.

The generative adversarial training scheme- Based on the above principles, the first GAN was introduced in 2014. It is important to understand the training process of generative learning. One key insight is the indirect training: this basically means that the generator is not trained to minimize the distance to a specific image, but just to fool the discriminator! This enables the model to learn in an unsupervised manner. Finally, pay attention that the ground truth when training the generator is 1 (like a real image) in the output, even though the examples are fake. This happens because our objective is just to fool D. The image below illustrates this process:

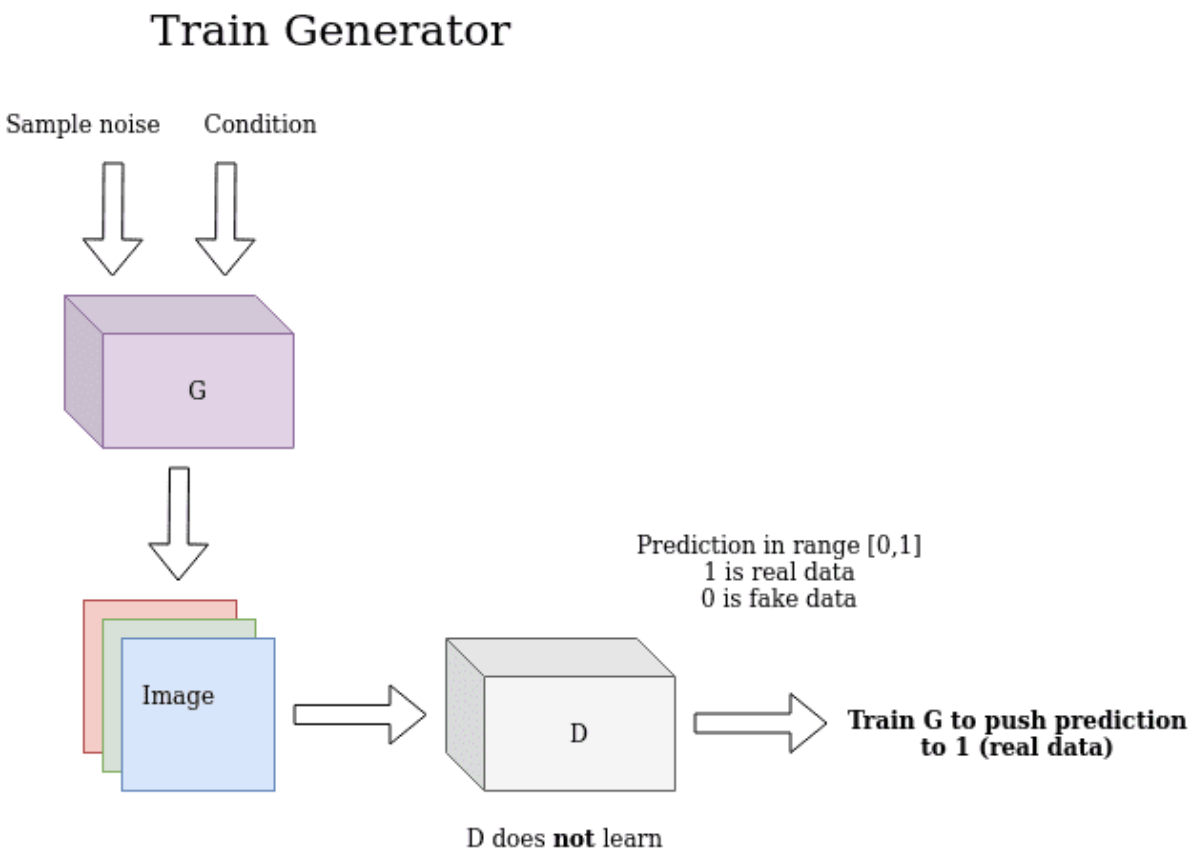

Fig 2.1 Generator training scheme of Vanilla GANs

On the other hand, the discriminator uses the real image and the generated to distinguish them, pushing the prediction of the real image to 1(real) and the prediction of the generated image to 0 (fake). In this stage, notice that the Generator is not trained.

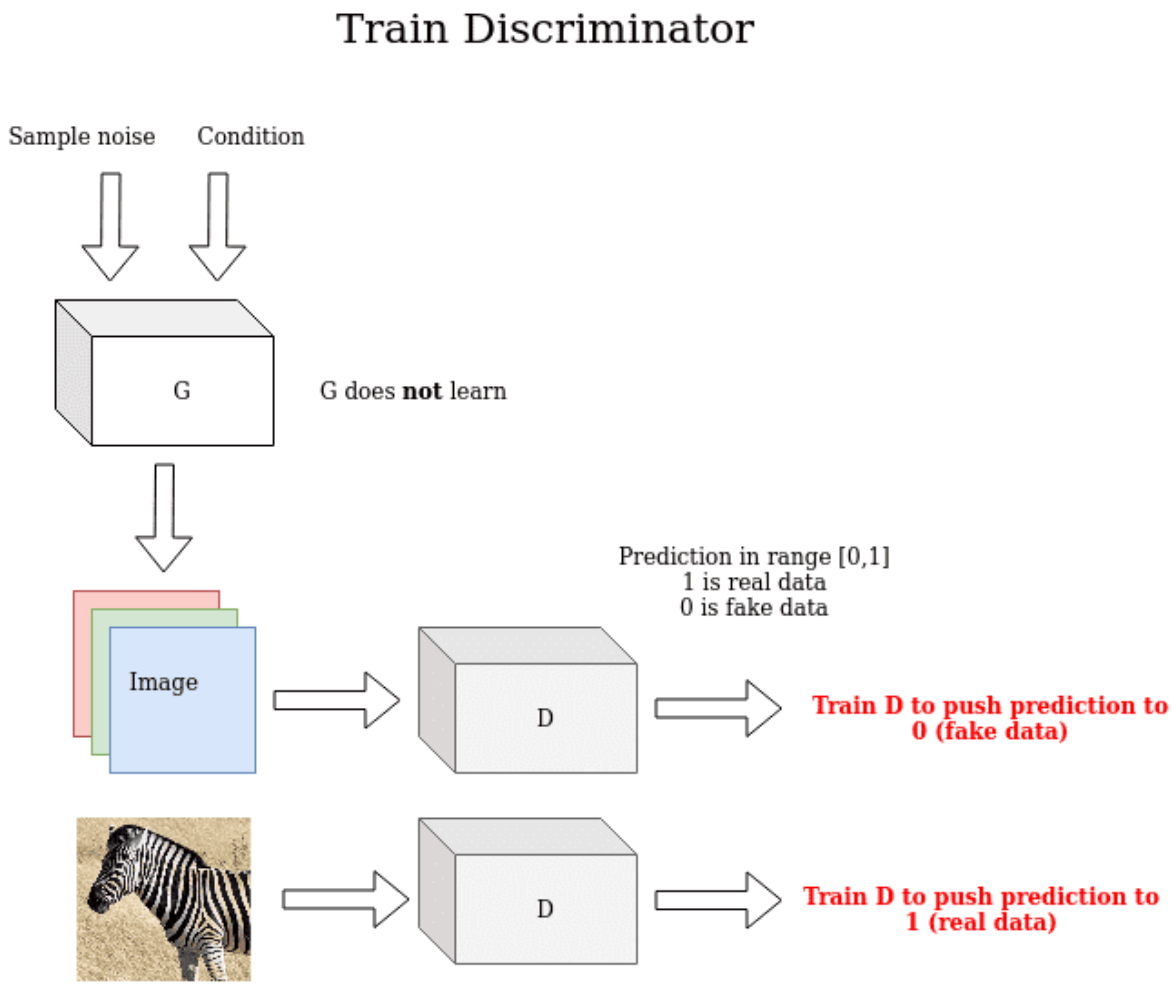

Fig 2.2 Discriminator training scheme of Vanilla GANs

The authors of GAN proposed an architecture consisting of 4 fully-connected layers for both G and D. Of course, G outputs a 1-D vector of the image and D outputs a scalar in the range (0,1).

GAN training scheme in action: Representation gives us power over ideas. It is nice to understand the math before seeing the code. So, given discriminator D, generator G, vale function V the fundamental two-player minimax game can be defined as (M are the samples):

$$\min_G \max_D V(G,D) = E_{\mathbf{x}\sim p_{data}(\mathbf{x})}[logD(\mathbf{x})] + E_{z\sim p_{fake}(z)}[log(1-D(G(z)))]$$

In practice, recalling the discete definition of the expected value, it looks like this:

$$L_D = \frac{1}{M}\sum_{i=1}^{M}[logD(\mathbf{x}^i) + log(1 - D(G(\mathbf{z}^i)))]$$

$$L_G = E_{z\sim p_{fake}(z)}[logD(G(z))] = \frac{1}{M}\sum_{i=1}^{M}[log(D(G(\mathbf{z}^i)))]$$

In fact, the discriminator performs gradient ascend, while the generator performs gradient descent. This can be observed by inspecting the differences in the generator part of the above equations. Note that, D needs to access both real and fake data, while G has no access to the real images.

2.2 Existing Solution

The rapid development of deep learning has accounted for recent exciting progress in image generation, especially the introduction of generative adversarial networks (GAN) [2]. Conditioning variables were then introduced to GAN [3, 4]. Related to our contextual GAN for joint images is deep image completion with perceptual and contextual losses [5]. Pre Trained with uncorrupted data, the G and D networks are trained to reconstruct a complete image. Their impressive examples show that even if a large region of a facial image is cropped from the input, the generated complete facial image looks very realistic. Another impressive work on image completion [4] is based on autoencoders with a standard reconstruction loss and an adversarial loss. Autoencoders have also been successfully applied to generating images from visual attributes [6].

Analogous to image completion, where the uncropped part of the image provides the proper context for facial image completion, in our sketch-to-image generation, the entire input sketch is regarded as the “uncropped context” for completing the entire

natural image part of the joint image. Another way of generating images from sketches requires a huge database from which images are retrieved. In [7], a database of sketch-photo pairs was collected, and deep learning was used to learn a joint embedding. The Sketchy Database [8] contributed a collection of sketch-photo pairs which were used to train a cross-domain CNN to embed them in the same space. In recent works, the mapping between sketches and images was studied in sketch-based image retrieval, where the sketch and the image were learned in separate networks. In, several triplet CNNs [9] were evaluated for measuring the similarity between sketches and photos. Triplet networks are used to learn joint embeddings. While classical representations were proposed to retrieve images from sketch queries [10, 11, 12], recent methods used sophisticated feature representations [13, 14]. Recent cross-domain embedding methods trained deep networks to learn a common feature space for sketches and 3D models [15], and images and 3D models. Siamese networks trained with contrastive loss [16] and triplet or ranking loss [17] were proposed. On the other hand, we do not require such sketch-photo collections or regard sketch and image as two separate domains, as they form the same joint image. This allows for an effective encoding for the “corrupted” joint image in the latent space and leads to stable training.

2.3 Hardware and Software Requirements

2.3.1 Hardware Requirements

Table 2.1 lists the hardware requirements of the system.

Table 2.1 Hardware Requirements

Operating System	Windows 10, MAC OS

2.3.2 Software Requirements

Table 2.2 lists the hardware requirements of the system.

Table 2.2 Software Requirements

Programming Language	Python HTML, CSS, JavaScript, Flask
Web Browser	Safari, Google Chrome, etc.

Chapter 3

Analysis

3.1 Functional Requirement

Functional Requirements includes the basic functionalities that the system offers but which one can see directly in the final product.

- Input must be an image of a sketch of a person's face.
- System must perform transformation of the input sketch to an output face image.
- Output must be a corresponding face image of the input sketch image.

3.2 Non-Functional Requirement

Non-functional Requirements includes the quality constraints that the system must satisfy in order to fulfil the functional requirements.

- Input must be in the form of one of these image file formats - .jpg, .jpeg or .png
- Transformation of the sketch image to the face image should be done within 30 seconds.
- The site should load in 2 seconds and be compatible with all web browsers such as Google Chrome, Firefox, Microsoft Edge etc.
- Output face image must be downloadable.

3.3 Proposed System

We propose a novel Contextual Generative Adversarial Network (cGAN) that tackles the challenges faced by the existing solutions for sketch-to-image generation. We rephrase the image generation problem as an image completion problem, wherein the sketch image provides a weak contextual constraint.

In conventional image completion, the corrupted part of an input image is completed using surrounding image content as context. A generative adversarial network is trained to learn the joint distribution and capture the inherent correspondence between a sketch and its corresponding image using the defined joint image. This approach encodes the “corrupted” joint image into the closest “uncorrupted” joint image in the latent space via back propagation, which can be used to predict and hence generate the output image part of the joint image.

'Inpainting' is generating an image from itself. One small portion of the input image is corrupted with a mask. This image, which is partially covered with a mask, as shown in Figure 3.1, is called the context. The model tries to generate a whole image with respect to this context. We have employed this concept but our input image consists of the sketch and the face image side by side. Say we have our sketch image to the left of our face image. We corrupt the right side of our input image, which consists of the face image, with a mask. Now, our model uses this partially masked image as the context and generates a completed image with respect to this context.

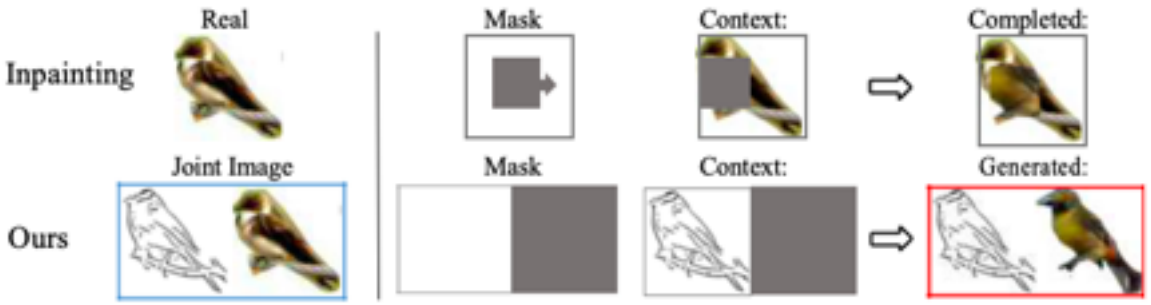

Fig 3.1 Contextual GANs Inpainting Methodology

3.3.1 Flow of Model

The Sketch-to-Image Transformation Model is created as depicted in Figure 3.2. First images are imported from the CUHK Face Sketch database, described in Section 3.3.2. Then image augmentation is performed on this dataset, described in Section 3.3.3. Next, generator and discriminator models are trained, and once training is finished, the trained model is saved.

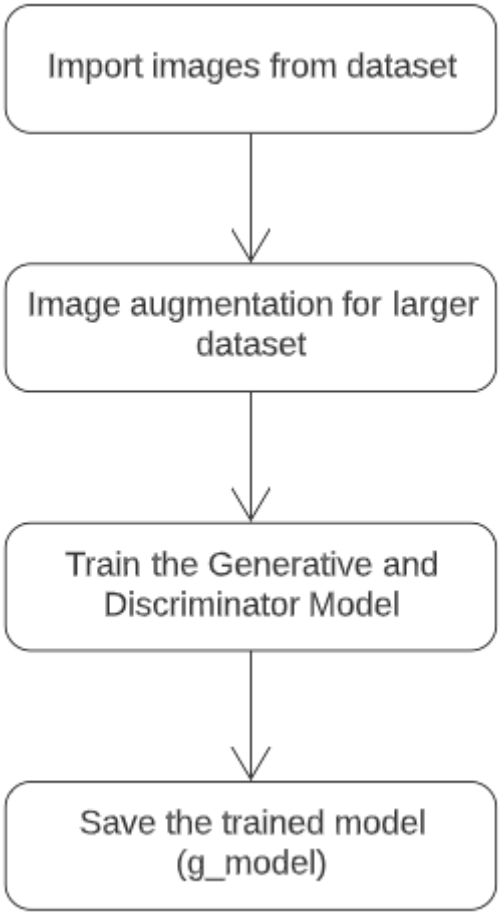

Fig 3.2 Flow of Sketch-to-Image Model Creation

The Sketch-to-Image Application starts with loading image from the user. This image then passes through the trained Sketch-to-Image transformation model. To further enhance the model, the image is passed through GFP-GANs and DeOldify, which are pre-built functions. Lastly, the face image is created and is sent to teh user over front end. This process is depicted in Figure 3.3.

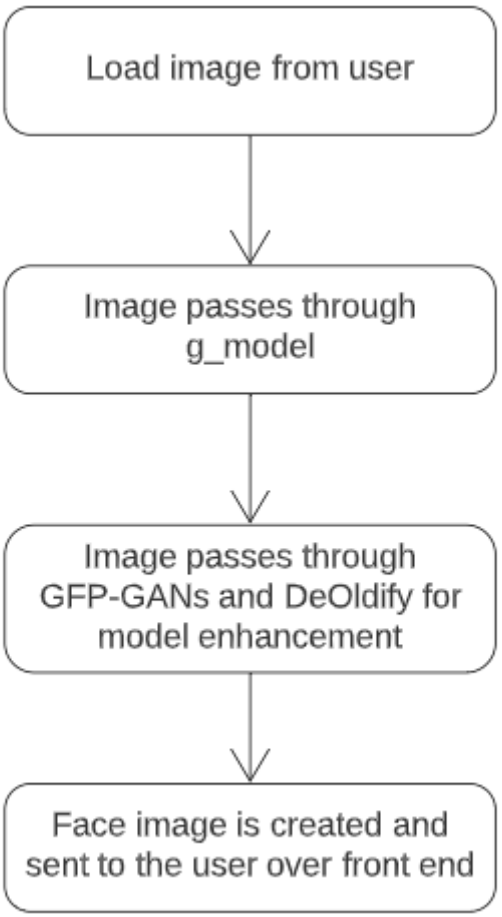

Fig 3.3 Flow of Sketch-to-Image Application

3.3.2 Dataset

We have used the CUHK Face Sketch database (CUFS) [18]. This database was originally made for research on face sketch synthesis and face sketch recognition. But, it is also suitable for our model.

It includes 188 faces from the Chinese University of Hong Kong (CUHK) student database, 123 faces from the AR database, and 295 faces from the XM2VTS database. There are 606 faces in total. For each face, there is a sketch drawn by an artist based

on a photo taken in a frontal pose, under normal lighting condition, and with a neutral expression. Figure 3.4 shows some instances of the dataset.

The dataset is divided as 88 faces for training and 100 faces for testing. For each sketch, there is a cropped sketch and a .dat file of the fiducial points of the sketch as well. For each photo, there is a cropped photo and a .dat file of the fiducial points of the photo as well.

Fig 3.4 CUHK Face Sketch Dataset

Since this dataset is not enough to train our model, we have performed image augmentation to create a larger chunk of data for our model to train on. Now, from 600 odd images, we have around 17,000 images to train our model on, and this is a considerable amount of data to ensure we have an accurate and properly trained model.

3.3.3 Pre-Processing Techniques

Data augmentation is a technique to increase the diversity of dataset without collecting any more real data but still improving model accuracy and preventing the model from overfitting. The goal of image augmentation is to artificially increase the size of your training image dataset by generating modified copies of the original images. OpenCV or Open Source Computer Vision is one of the most widely used Python tools for computer vision and image processing tasks. We have pre-processed out data by performing image augmentation using OpenCV.

Image Augmentation may include Horizontal or Veritical Shift or Translation, Brightness, Zoom, Channel Shift, Horizontal or Vertical Flip, Rotation, Fill Mode (Nearest, Reflect, Wrap, Constant), etc.

Chapter 4

Design

4.1 Visual Diagram Design

UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems. We prepare UML diagrams to understand the system in a better and simple way. A single diagram is not enough to cover all the aspects of the system. UML defines various kinds of diagrams to cover most of the aspects of a system.

There are two broad categories of diagrams and they are again divided into subcategories − Structural Diagrams and Behavioral Diagrams. The structural diagrams represent the static aspect of the system. These static aspects represent those parts of a diagram, which forms the main structure and are therefore stable. These static parts are represented by classes, interfaces, objects, components, and nodes. The four structural diagrams are − Class diagram, Object diagram, Component diagram,

and Deployment diagram. Behavioral diagrams basically capture the dynamic aspect of a system. Dynamic aspect can be further described as the changing/moving parts of a system. UML has the following five types of behavioral diagrams − Use case diagram, Sequence diagram, Collaboration diagram, Statechart diagram, and Activity diagram [19].

4.1.1 Use Case Diagram

Use case diagrams are used to gather system design requirements of a system. When a system is analyzed to gather its functionalities, use cases are prepared and actors are identified. When the initial task is complete, use case diagrams are modelled to present the outside view. Use case diagrams are − used to gather the requirements of a system, used to get an outside view of a system, identify the external and internal factors influencing the system, and show the interaction among the actors [19].

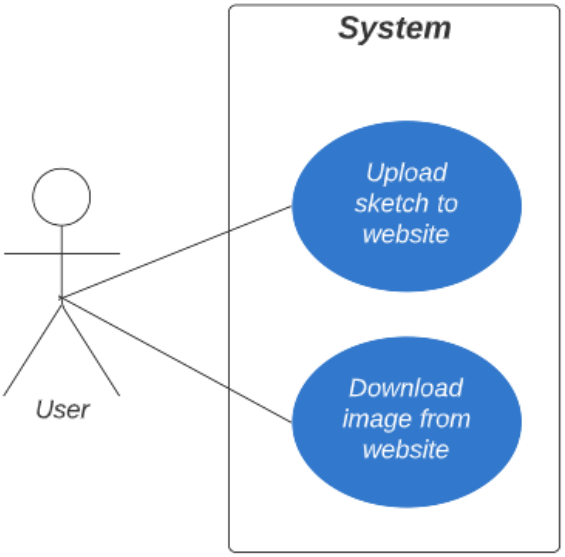

Fig 4.1 Use Case Diagram

Table 4.1 Use Cases

Actors	Use Cases
User	1. Upload a sketch to website 2. Download image from website

4.1.2 State Diagram

State diagram defines the different states of an object during its lifetime, from creation to termination, and the flow of control from one state to another. States are defined as a condition in which an object exists and it changes when some event is triggered. State diagrams are useful in modeling the reactive systems - those that respond to external or internal events [19].

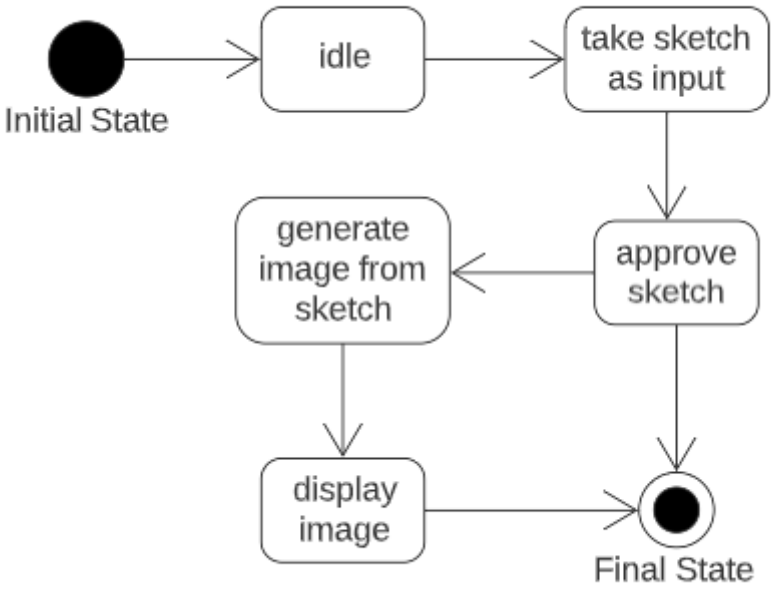

Fig 4.2 State Diagram

4.1.3 Activity Diagram

An activity is an operation of the system. Activity diagrams are used for visualizing the dynamic nature of a system, and to construct the executable system by using forward and reverse engineering techniques. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc. They show different flows such as parallel, branched, and single [19].

Activities:

- User uploads a sketch
- System approves the sketch
- System transforms the sketch to an image
- System presents final image
- User downloads the final image

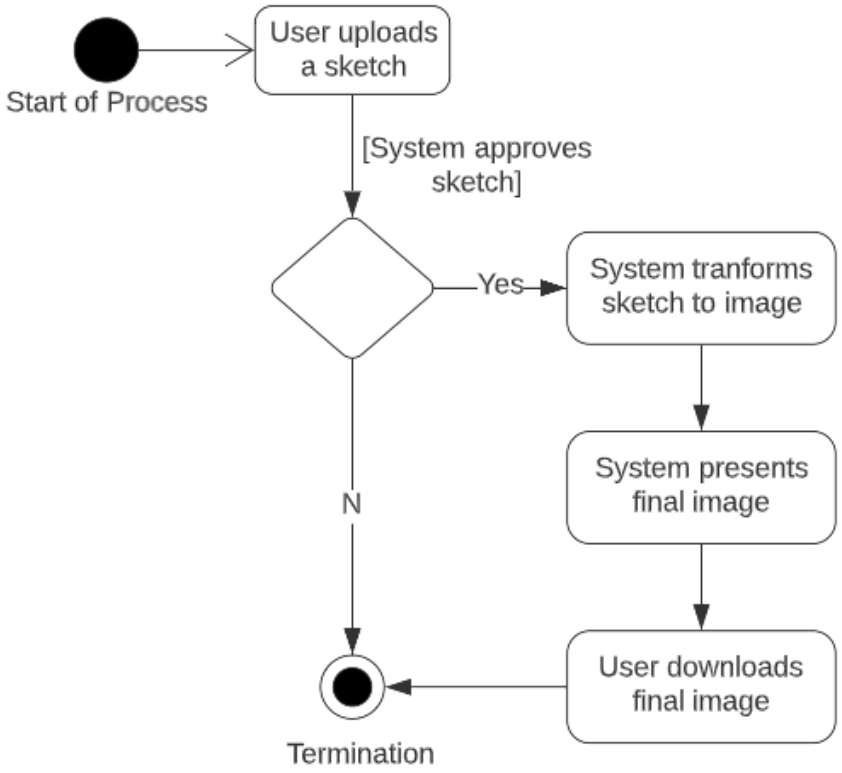

Fig 4.3 Activity Diagram

4.2 GUI Design

Figure 4.4 shows the User Interface for the Sketch-to-Face Model. It includes an upload button for the user to upload the sketch.

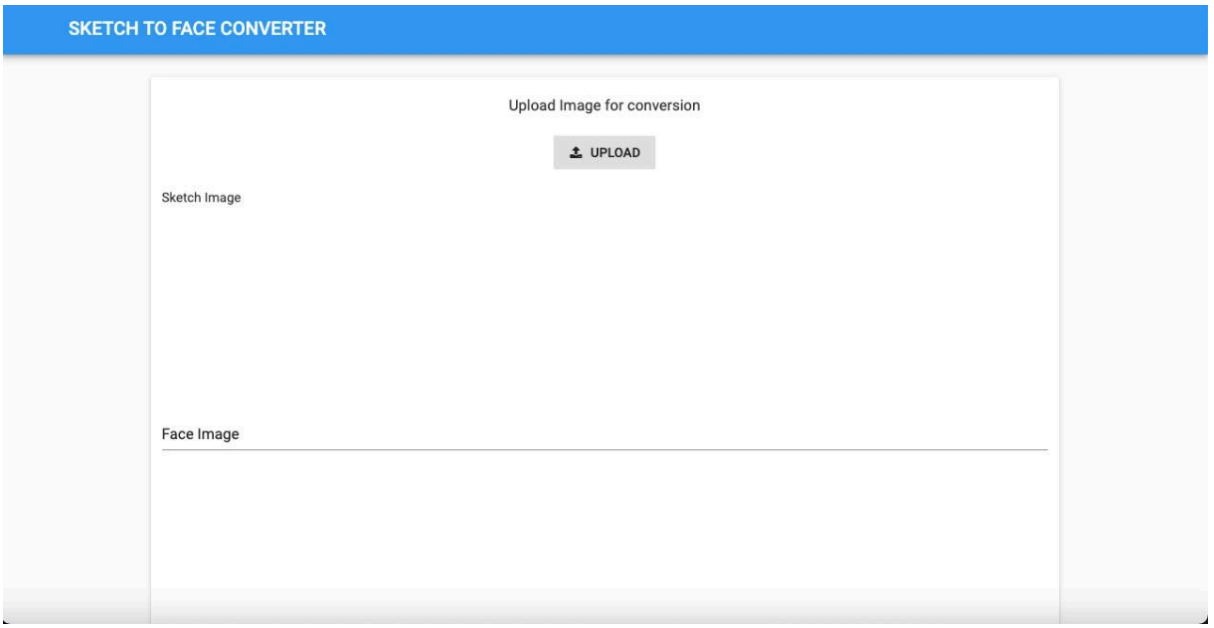

Fig 4.4 GUI to upload the sketch

Figure 4.5 shows how to upload the sketch from the user's local device.

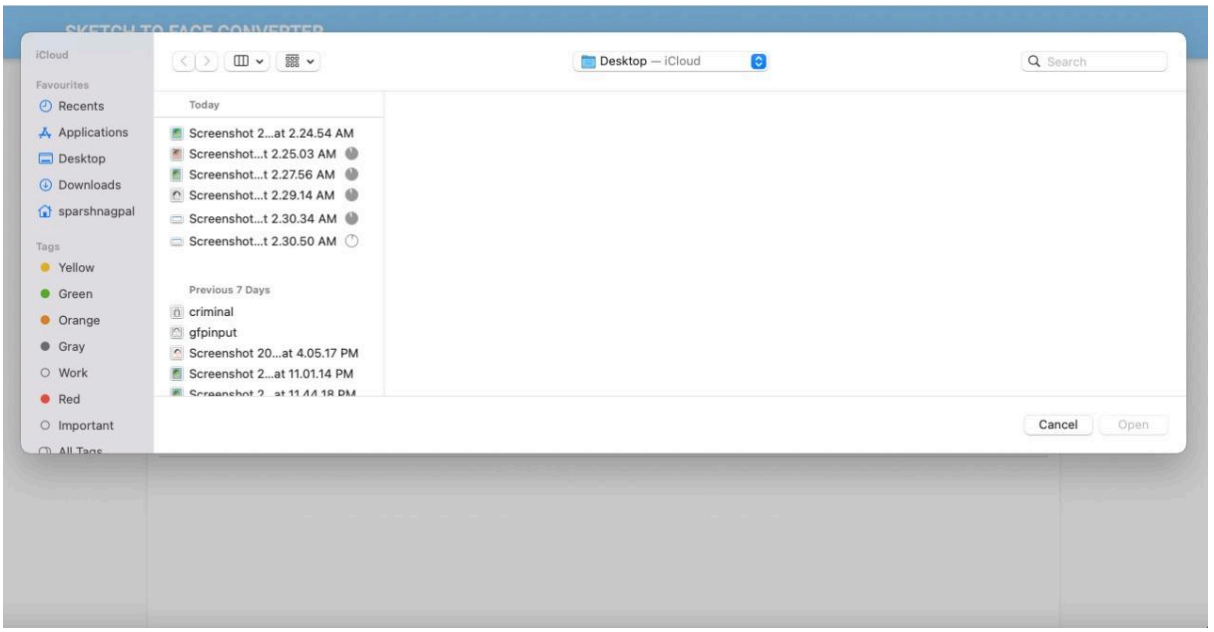

Fig 4.5 Uploading from your local device

Figure 4.6 shows the output face image after transformation.

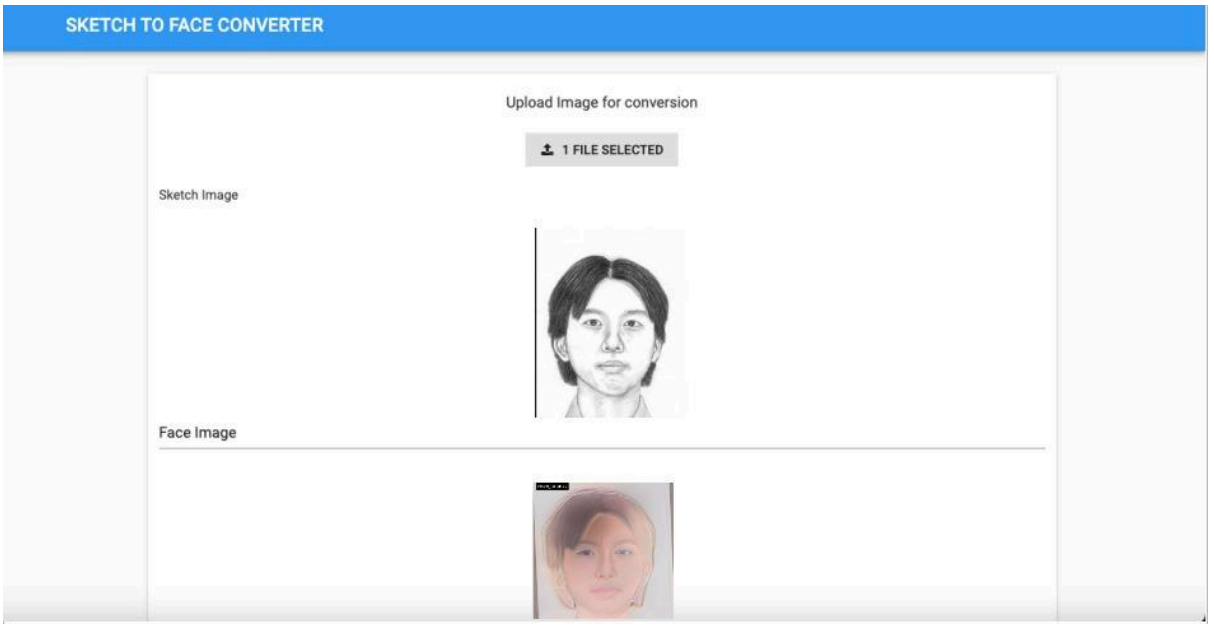

Fig 4.6 Output Image

Chapter 5

Implementation

5.1 Implementation

5.1.1 Contextual GANs

GANs have two networks, a generator network and a discriminator network. The generative adversarial networks use the confrontation of two neural networks as the training criterion and use back propagation to train. In the training process, the traditional Markov chain method is abandoned, and there is no complicated variational lower bound. Therefore, the training efficiency and training difficulty of generating model have been improved [20]. GANs can directly sample and infer new samples in the encoding process, which improves the coding efficiency of the original samples significantly.

Contextual GAN consists of training stage and completion stage. The trainings stage is the same as the traditional GAN training except that the training stage has a sample of joint images. After training, we learn a generative network G that achieves the objective of reproducing the joint image data distribution.

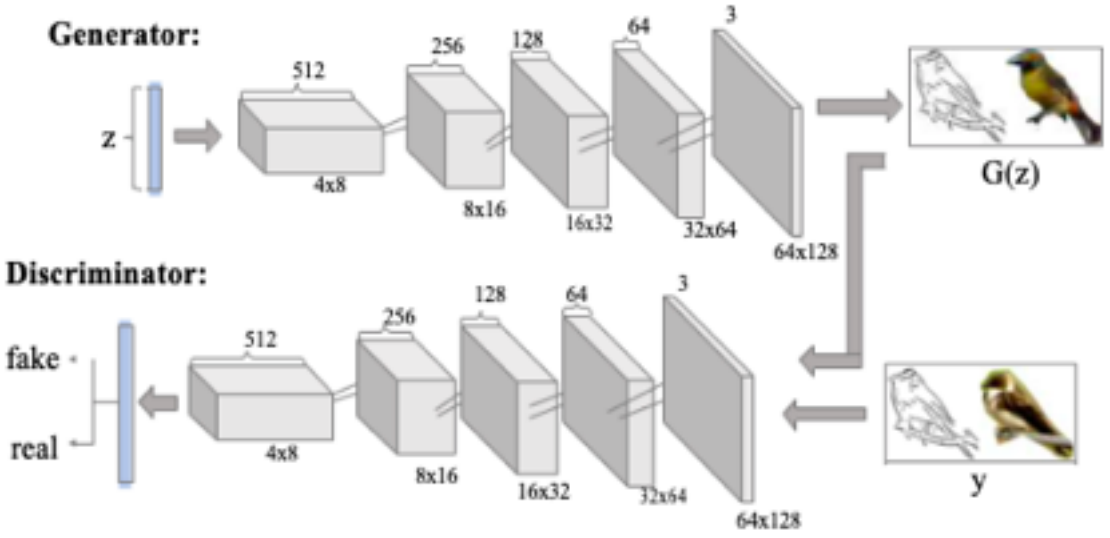

Fig 5.1 Generator Discriminator Architecture for Contextual GANs

We have also used pre-built functions for image enhancements, such as GFP-GANs and DeOldify. GFP-GAN is a generative adversarial network for blind face restoration that leverages a generative facial prior (GFP). This Generative Facial Prior (GFP) is incorporated into the face restoration process via channel-split spatial feature transform layers, which allow for a good balance between realness and fidelity [21]. DeOldify is used to colorize and restore old images and film footage.

5.2 Result and Evaluation

5.2.1 Results

We have able able to generate real life like images through image translation method using Contextual GANs. To measure the performance, we used metrics like SSIM score and L2-regularization score. SSIM score was tested as an approximate of 0.77 and L2-regularization score as 93.

Fig 5.2 Result of Sketch-to-Image Transformation Model

5.2.2 Evaluation

- **Structural Similarity Index**

Structural Similarity Index (SSIM) is a a metric used to measure the similarity between two given images. This metric calculates the Structural Similarity Index between 2 given images which is a value between -1 and +1. A value of +1 indicates that the 2 given images are very similar or the same while a value of -1 indicates the 2 given images are very different. The SSIM metric extracts 3 key features from an image - Luminance, Contrast and Structure - on the basis of which the comparison between the two images is performed [22].

Luminance comparison function is given by l(x,y) = 2xy + C1x2 + y2 + C1, where C1 is a constant to ensure stability when the denominator becomes 0, and C1=(K1L)2. Contrast comparison function is given by c(x,y) = 2xy + C2x2 + y2+ C2, where C2= (K1L)2. Structure comparison function is given by s(x,y) = xy + C3 xy + C3, where xy= 1N-1i=1N(xi-x)(yi-y). SSIM Score is given by SSIM(x,y) = [l(x,y)]. [c(x,y)] . [s(x,y)], where $\alpha > 0$, $\beta > 0$, $\gamma > 0$ denote the relative importance of each of the metrics. To simplify the expression, if we assume, $\alpha = \beta = \gamma = 1$ and C3 = C2/2, we can get, SSIM(x,y) =(2xy + C1)(2xy + C2)(x2 + y2 + C1)(x2 + y2+ C2)

- **L2 Regularization**

Overfitting is a significant issue in the field of data science that needs to handled carefully in order to build a robust and accurate model. Overfitting arises when a model tries to fit the training data so well that it cannot generalize to new observations. Overfit models seem to be outstanding on training data but performs poor on new, previously unseen observations. The main reason of overfitting is model complexity. Thus, we can prevent a model from overfitting by regularization. Regularization controls the model complexity by penalizing higher terms in the model. If a regularization terms is added, the model tries to minimize both loss and complexity of model. There are two commonly used regularization techniques - L1 and L2 regularization. The two main reasons that cause a model to be complex are - the total number of features (handled by L1 regularization), or the weights of features (handled by L2 regularization).

If we take the model complexity as a function of weights, the complexity of a feature is proportinal to the absolute value of its weight. $y = w_1 x_1 + \ldots\ldots + w_n x_n$

$$L2\ Regularization\ terms = w_1^2 + w_2^2 + \ldots\ldots + w_n^2$$

L2 regularization forces weights toward zero but it does not make them exactly zero. L2 regularization acts like a force that removes a small percentage of weights at each iteration. Therefore, weights will never be equal to zero.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

With the help of contextual GANs which formulate the sketch-to-image generation as the joint image completion problem, we implemented this model to create a realistic (or as close as possible) photograph of a person from the given input sketch. Our model is beneficial to use for criminal investigation as it replaces the old methodologies and gives us a more detailed and accurate output compared to a sketch. It generates images with an accurate facial feature which is easy for a human brain to comprehend and gives us the details and nuances of a face that is not possible to understand with the help of a sketch. Overall the model gave a similarity score (SSIM) of 0.77 and an L2-regularization score of 93. It also includes an easy-to-use user interface where the user can upload a sketch and after the processing of the model, they can see the desired realistic photograph as an output.

6.2 Future Work

The Sketch-to-Image model is a solid base for further advancements to be made on it. One consideration to be made in the future would be to use a more racially diverse dataset. The CUHK Face Sketch Dataset includes images of students of the Chinese University of Hong Kong and so we can expect most of them to be Chinese. As a result, our model is well trained to recognize and predict Chinese facial features, and provides accurate images from the given sketch. However, since different races have different dominant facial features, we cannot predict how well this model can predict those. Thus, to make the model universally usable with a higher accuracy, in the future, we must train it on a more racially diverse dataset. Other than this, training the model on a much larger dataset should also improve the model accuracy.

References

[1] Alexey Kurakin, Ian Goodfellow, Samy Bengio. (2016) Adversarial Machine Learning at Scale. In ICLR.

[2] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). Generative adversarial nets. In Advances in neural information processing systems (pp. 2672-2680).

[3] Oord, A.v.d., Kalchbrenner, N., Vinyals, O., Espeholt, L., Graves, A., Kavukcuoglu, K.: Conditional image generation with pixelcnn decoders. In: Proceedings of the 30th International Conference on Neural Information Processing Systems. pp. 4797–4805. Curran Associates Inc. (2016)

[4] Pathak, D., Kr¨ahenb¨uhl, P., Donahue, J., Darrell, T., Efros, A.: Context encoders: Feature learning by inpainting. In: CVPR (2016)

[5] Yeh, R.A., Chen, C., Lim, T.Y., Schwing, A.G., Hasegawa-Johnson, M., Do, M.N.: Semantic image inpainting with deep generative models. In: CVPR (2017)

[6] Yan, X., Yang, J., Sohn, K., Lee, H.: Attribute2image: Conditional image generation from visual attributes. In: European Conference on Computer Vision. pp. 776–791. Springer (2016)

[7] Yu, Q., Liu, F., SonG, Y.Z., Xiang, T., Hospedales, T., Loy, C.C.: Sketch me that shoe. In: CVPR (2016)

[8] Sangkloy, P., Burnell, N., Ham, C., Hays, J.: The sketchy database: Learning to retrieve badly drawn bunnies. ACM TOG (2016)

[9] Bui, T., Ribeiro, L., Ponti, M., Collomosse, J.P.: Generalisation and sharing in triplet convnets for sketch based visual search. CoRR abs/1611.05301 (2016), http://arxiv.org/abs/1611.05301

[10] Bimbo, A.D., Pala, P.: Visual image retrieval by elastic matching of user sketches. IEEE Trans. Pattern Anal. Mach. Intell. 19(2), 121–132 (1997). https://doi.org/10.1109/34.574790, http://dx.doi.org/10.1109/34.574790

[11] Kato, T., Kurita, T., Otsu, N., Hirata, K.: A sketch retrieval method for full color image database-query by visual example. In: ICPR. pp. 530–533 (Aug 1992). https://doi.org/10.1109/ICPR.1992.201616

[12] Sclaroff, S.: Deformable prototypes for encoding shape categories in image databases. PR 30(4), 627–641 (1997). https://doi.org/10.1016/S0031-3203(96)00108-2, http://dx.doi.org/10.1016/S0031-3203(96)00108-2

[13] Cao, Y., Wang, C., Zhang, L., Zhang, L.: Edgel index for large scale sketch-based image search. In: IEEE CVPR. pp. 761–768 (2011). https://doi.org/10.1109/CVPR.2011.5995460, http://dx.doi.org/10.1109/CVPR.2011.5995460

[14] Shrivastava, A., Malisiewicz, T., Gupta, A., Efros, A.A.: Data-driven visual similarity for cross-domain image matching. ACM TOG 30(6) (2011)

[15] Wang, F., Kang, L., Li, Y.: Sketch-based 3d shape retrieval using convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1875–1883 (2015)

[16] Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with an application to face verification. In: CVPR. vol. 1, pp. 539–546 vol. 1 (June 2005). https://doi.org/10.1109/CVPR.2005.202

[17] Wang, J., Song, Y., Leung, T., Rosenberg, C., Wang, J., Philbin, J., Chen, B., Wu, Y.: Learning fine-grained image similarity with deep ranking. In: CVPR. pp. 1386–1393 (2014)

[18] CUHK Face Sketch Database (CUFS) [Online]. Available: http://mmlab.ie.cuhk.edu.hk/archive/facesketch.html

[19] “Unified Modelling Language” tutorialspoint.com. https://www.tutorialspoint.com/uml/index.htm (accessed Apr. 19, 2022).

[20] Hu, M., Guo, J. Facial attribute-controlled sketch-to-image translation with generative adversarial networks. *J Image Video Proc.* 2020, 2 (2020).

[21] Wang et al. in Towards Real-World Blind Face Restoration with Generative Facial Prior. Available: https://paperswithcode.com/method/gfp-gan

[22] “All about Structural Similarity Index (SSIM)”. Pranjal Datta. medium.com https://medium.com/srm-mic/all-about-structural-similarity-index-ssim-theory-code-in-pytorch-6551b455541e (accessed Apr. 19, 2022).

[23] “L1 and L2 Regularization — Explained”. Soner Yıldırım. https://towardsdatascience.com/l1-and-l2-regularization-explained-874c3b03f668 (accessed Apr. 19, 2022).

Acknowledgement

We would like to express our gratitude and thanks to **Prof. Vaibhav Ambhire** for his valuable guidance and help. We are indebted for his guidance and constant supervision as well as for providing necessary information regarding the project. We would like to express our greatest appreciation to our principal **Dr. G.T. Thampi** and head of the department **Dr. Tanuja Sarode** for their encouragement and tremendous support. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of the project.

Sparsh Nagpal

Aryan Nayak

Rachaell Nihalaani