CS550 - Machine Learning

Report on
Few-shot Knowledge Transfer for Fine-grained Cartoon Face Generation

Group Name: ***R53***
Group Members: *Dhruv Deshmukh(11940380)* *and* *Anupam Kumar(11940160)*

Abstract

This project focuses on generating fine-grained cartoon images given real photos of a person. In the database, there is an imbalance in a particular group of faces and this limits the capabilities of the model to generate the cartoon faces for that group. To improve the results for the minority group a different training strategy is proposed consisting of two stages. In this strategy, certain layer parameters of the model are specific to a group and others are shared among all groups. In the first training stage, the majority group is used for training to get the baseline values of all parameters. Then the model is trained on the minority groups to get the proper values of group-specific parameters. In this way, knowledge is transferred from the majority group to minority groups.

1. Introduction

Cartoon faces are greatly used in our day-to-day lives. It is often spotted in social media platforms like Instagram, Facebook, Twitter, WhatsApp, Snapchat, etc as profile pictures, posts, stories, and many more. Cartoon images are also used in memes and stickers. A professional artist takes hours to make a cartoon image from a real image. We aim to design a Machine Learning model which can perform the same task of converting a real-face image into a realistic cartoon image. For this, we generate an image-to-image translation model.

Image to image translation has gained a lot of importance in the field of **Machine Learning** and **Computer Vision.** It was first introduced as a model which utilized the **Generative Adversarial Network** (GAN) to learn mapping functions from paired training samples.

This project focuses on “Face to Cartoon” conversion. There are several ways to do this, but we perform this conversion by dividing the faces into groups based on some key characteristics, and hence, we have created the groups of young women, young men, children, and the elderly. The cartoon faces of young women are seen to have bigger eyes with eyelashes whereas the cartoon faces of men do not have eyelashes and comparatively smaller eyes. The elderly’s faces are usually wrinkled whereas the lower age group people's (kid’s) faces are not wrinkled. This is the basic concept used in our model to improve its performance. Finally, we assume that although the appearance of faces of different groups may be different, all the faces still share a common cartoon style. Consequently, we can train an image translation model for one common group and then transfer knowledge to another group with only a few samples. Therefore, we design a multi-branch image translation network with fine-grained face generation where the main branch learns to translate images from the common group and maintain the distribution of the shared feature space while other branches learn specific characteristics for each rare group.

2. Problem Definition

The main problem is translation images from the real photo domain(X) to the cartoon image domain(Y). In this as well we further classify the photos into four groups: young women, young men, children, and elderly. Let us number the class as 0,1,2,3 respectively. We have sufficient data for group 0 while the data is less for the other three groups. Hence the majority group is 0 and others are minority groups. As proposed our task is to train on the majority group first and then transfer the knowledge to minority groups.

3. Objective

The main objective is to obtain good results for all groups 0,1,2,3 and not just the majority group. But due to the lack of sufficient data for all 4 groups the strategy of knowledge transfer is used to learn the main parameters using the majority group and then transferring the parameters to other groups.

To do the photo to cartoon conversion image to image translation is needed. This is achieved by using a variational autoencoder.

Also, the autoencoder is used in a generator of GAN to train it with a discriminator that identifies real and fake cartoon images. This helps the autoencoder to learn the style of the cartoon.
The type of GAN used is CycleGAN which enables unsupervised learning using unpaired images of real faces and cartoon faces. In this, a better model called U-GAT-IT is adopted which uses better normalization functions.
Also, certain losses like Face ID loss have been added to keep a consistent styling.

4. Technology Used

The technology used can be categorized into three subcategories:

1. Generative Adversarial Networks
2. Image-to-Image Translation
3. Mobile Face-net
4. Few-shot Image-to-Image Translation

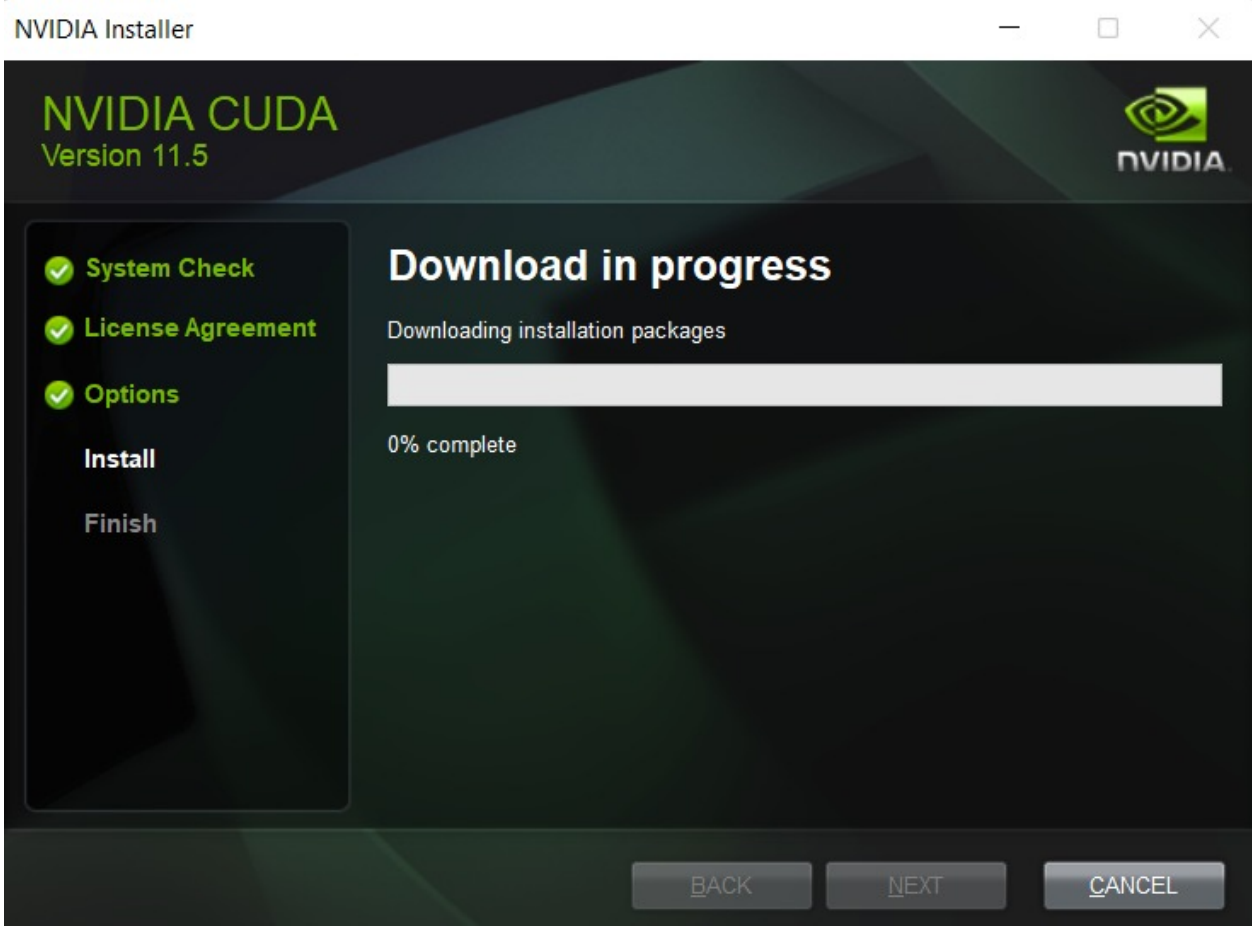
The main framework used to code the models is PyTorch. Besides this OpenCV has been used for preprocessing the data and also Numpy for data manipulation and linear algebra tasks. The models are explained in detail in the model's attention-based used section.

5. Problems Faced

The model requires a few training examples for minority groups as well. There was a problem finding a segregated dataset having separate classes for each group. Also, the minority groups had a shortage of cartoon images. The images given in the paper were used to reduce the shortage.
The images from different sources can have different dimensions which can cause a problem for the model. To solve this problem preprocessing is done to make all the training images uniform.
Finally, the training was not possible on our local machines due to the lack of enough memory resources on the GPU. This was after reducing the batch size to the minimum possible value.

```
RuntimeError: CUDA out of memory. Tried to allocate 20.00 MiB (GPU 0; 2.00 GiB total capacity; 908.98 MiB already alloca
ted; 18.82 MiB free; 962.00 MiB reserved in total by PyTorch)
```

Softwares like CUDA and CUDA-enabled Pytorch have a large download size and this caused a few problems as we were using mobile data for this. Also, the installation of Nvidia CUDA would not progress as the download process began even after leaving it straight for hours. This problem was resolved when we ran the model on Google Collab using the remote GPU of the server.



6. Datasets Used

For cartoon images, the dataset provided in the GitHub repo was used which consisted of 193 images in trainB and 10 images in testB. For real images, we used https://www.kaggle.com/greatgamedota/ffhq-face-data-set dataset. It consists of 70000 images, we used only 252 for quick training in trainA and 9 in testA. The cartoon mostly consists of young women while real photos consist of all types of data. Additionally, for testing, we used random images from the internet.

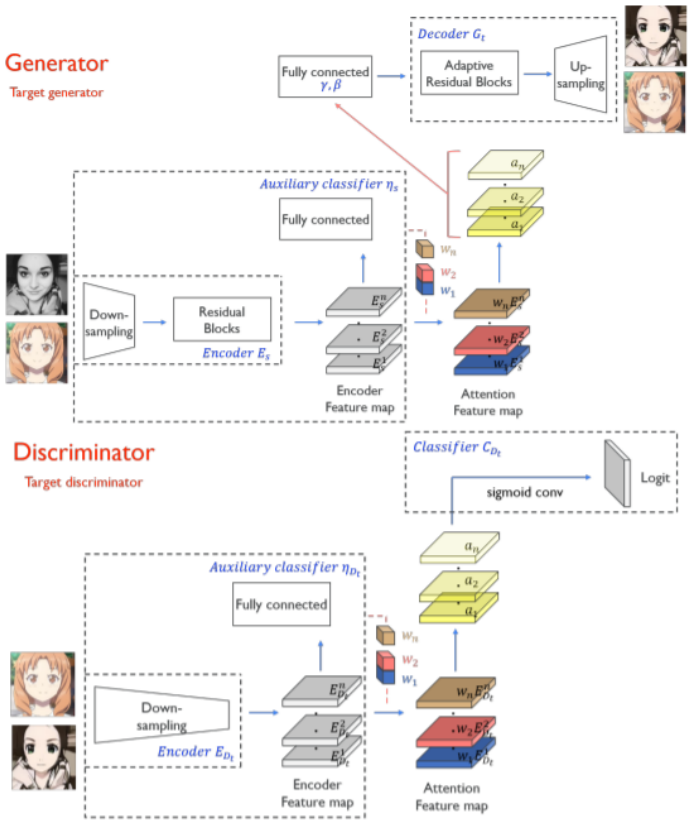
7. Models Used

The main model is the U-GAT-IT. It is a better version of CycleGAN and can be used for unsupervised learning using unpaired data. The CycleGAN tends to be unstable. U-GAT-IT overcomes this problem by using AdaLIN(Adaptive Layer-Instance Normalization). It chooses a proper proportion between Adaptive Normalization and Layer Normalization to control the Class Activation Map(CAM). CAM is an attention-based module that is used to focus on the important features in the image to ensure proper real to cartoon translation.

The other model is the MobileFaceNet which is used to calculate the Face ID loss. The models are discussed below:

U-GAT-IT

Here is the architecture of U-GAT-IT:



The model consists of a Generator and a Discriminator. The Generator itself consists of a Variational AutoEncoder. The CAM module is implemented within both the generator and the discriminator. Also, each of them has an Auxiliary classifier. The Residual Blocks are equipped with the AdaLIN. The AdaLIN function helps the attention-guided model to flexibly control the amount of change in shape and texture. As a result, the model, without modifying the model architecture or the hyper-parameters, can perform image translation tasks not only requiring holistic changes but also requiring large shape changes.

The AdaLIN function is:

$$AdaLIN(a,\gamma,\beta) = \gamma \cdot (\rho \cdot \hat{a_I} + (1-\rho) \cdot \hat{a_L}) + \beta,$$
$$\hat{a_I} = \frac{a-\mu_I}{\sqrt{\sigma_I^2+\epsilon}}, \hat{a_L} = \frac{a-\mu_L}{\sqrt{\sigma_L^2+\epsilon}},$$
$$\rho \leftarrow clip_{[0,1]}(\rho - \tau\Delta\rho)$$

where μI , μL and σI , σL are channel-wise, layer-wise mean and standard deviation respectively, γ and β are parameters generated by the fully connected layer, τ is the learning rate and $\Delta\rho$ indicates the parameter update vector (e.g., the gradient) determined by the optimizer.

As the goal of the attention model is to learn the important features of real photos and work on translating them in a better manner the auxiliary classifier is designed to learn the weight of the kth feature map in the encoder. Hence a weight is assigned to each feature map of the encoder. The activation feature map can be obtained by simply multiplying the learned weight by the feature map itself.
The generator takes in a pair of unrelated real and cartoon photos and then converts the real photo to a cartoon. This pair is then passed to the discriminator.

The discriminator consists of an encoder, an auxiliary classifier and the final classifier that distinguishes between real and fake cartoon images. The discriminator also uses attention maps to focus on regions that are critical to distinguish between the real and fake cartoon images. The method used is the same as above to learn the weights of different layers in the discriminator. The map takes in the pair from the generator and tries to discriminate whether there is a fake cartoon image in the pair.

The following losses are used to train the model:
Adversarial loss: Employed to match the distribution of the translated images to the target image distribution:

$$L^{s\to t}_{lsgan} = (E_{x\sim Xt}\,[(D_t(x))^2\,] + E_{x\sim Xs}\,[(1 - D_t(G_{s\to t}(x)))^2\,])$$

Cycle loss To alleviate the mode collapse problem, a cycle consistency constraint is applied to the generator. Given an image $x \in Xs$, after the sequential translations of x from Xs to Xt and from Xt to Xs, the image should be successfully translated back to the original domain:

$$L^{s\to t}_{cycle} = E_{x\sim Xs}\,[|x - G_{t\to s}(G_{s\to t}(x)))|_1]$$

Identity loss To ensure that the colour distributions of the input image and output image are similar, we apply an identity consistency constraint to the generator. Given an image $x \in Xt$, after the translation of x using Gs→t, the image should not change.

$$L^{s\to t}_{identity} = E_{x\sim Xt}\,[|x - G_{s\to t}(x)|_1]$$

CAM loss By exploiting the information from the auxiliary classifiers η_s and η_{Dt} , given an image $x \in \{X_s, X_t\}$. $G_{s\to t}$ and D_t get to know where they need to improve or what makes the most difference between two domains in the current state:

$L^{s\to t}_{cam} = -(E_{x\sim Xs}[\log(\eta_s(x))] + E_{x\sim Xt}[\log(1-\eta_s(x))])$

$L^{Dt}_{cam} = E_{x\sim Xt}[(\eta_{Dt}(x))2\,] + E_{x\sim Xs}[(1-\eta_{Dt}(G_{s\to t}(x))^2\,]$.

To implement the face ID loss the MobileFaceNet is used along with the ArcFace loss. The architecture of the MobileFaceNet is :

Input	Operator	*t*	*c*	*n*	*s*
$112^2 \times 3$	conv3x3	-	64	1	2
$56^2 \times 64$	depthwise conv3x3	-	64	1	1
$56^2 \times 64$	bottleneck	2	64	5	2
$28^2 \times 64$	bottleneck	4	128	1	2
$14^2 \times 128$	bottleneck	2	128	6	1
$14^2 \times 128$	bottleneck	4	128	1	2
$7^2 \times 128$	bottleneck	2	128	2	1
$7^2 \times 128$	conv1x1	-	512	1	1
$7^2 \times 512$	linear GDConv7x7	-	512	1	1
$1^2 \times 512$	linear conv1x1	-	128	1	1

The model is inspired by MobileNetV2 but has been adjusted for mobile devices. The arc face loss is:

$$L_3 = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s(\cos(\theta_{y_i}+m))}}{e^{s(\cos(\theta_{y_i}+m))}+\sum_{j=1,j\neq y_i}^{n}e^{s\cos\theta_j}}$$

This is used instead of normal softmax loss in the classification layer to give better classification results. Now this model is used to extract face features from the real and cartoon images and then the cosine distance between these features has been defined as the **Face ID loss**:

$L_{face} = E_{x\sim Xs}[1-\cos(F(x), F(T^i_{s\to t}(x)))] + E_{x\sim Xt}[1-\cos(F(x), F(T^i_{t\to s}(x)))]$

$E_{s\to t}$ is the encoder, $G_{s\to t}$ is the decoder, D_t is the discriminator for target domain. $T^i_{s\to t}$ is the image translation function for the i th group and F is the face recognition model.

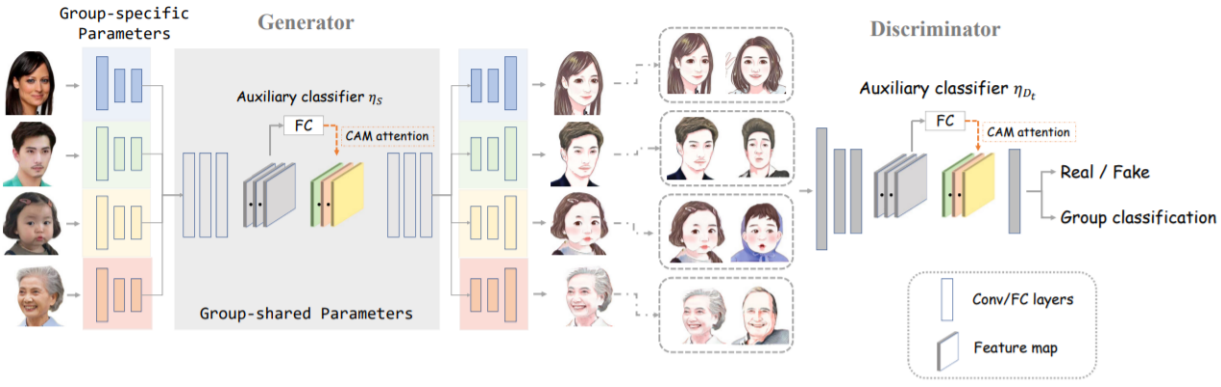
Finally, the few-shot knowledge transfer is used.

Few-shot Image-to-Image Translation is used in our model to map images in a given class to an analogous image in a different class, which is previously unseen during training. More precisely, we assure a group with sufficient training data is available and this group is used to transfer knowledge to other groups which contain fewer training samples.

8. Implementation

During the training, we obtain the real-face images and the corresponding cartoon-face images from them and denote them as domains X, Y respectively. Now, we train a generator that learns to map between the two domains for the defined multi-group. The real-face images are passed in the **Autoencoder** which comprises Encoder and Decoder consisting of several upsampling and downsampling blocks and the decoder regenerates cartoon-face images. The generator also comprises Class Activation Map (CAM) which helps the model to focus on more important regions. This image is further passed into the Discriminator to distinguish between real and fake images.

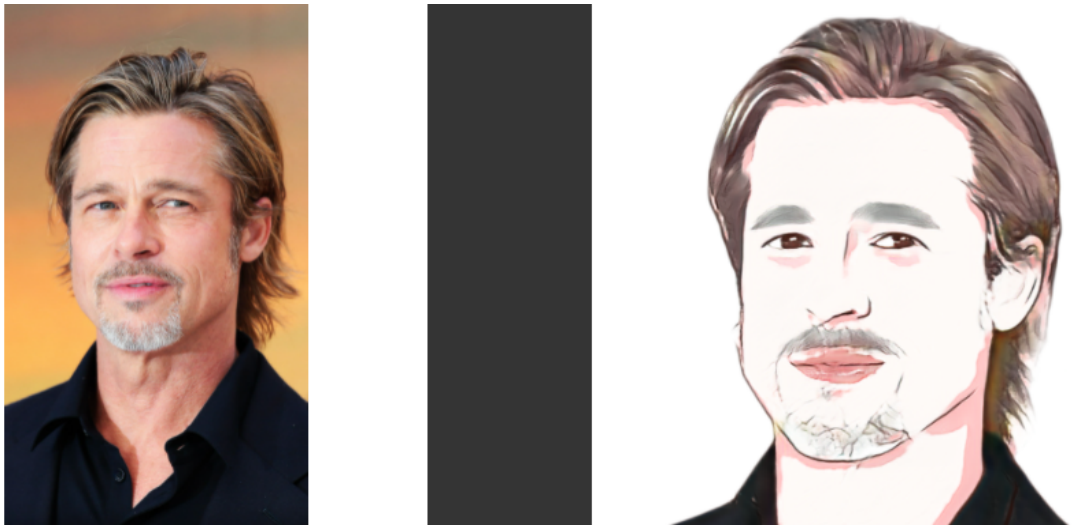
This training is done specifically for group 0 (young women) and at the end of the training, we transfer this knowledge (**Knowledge Transfer**) by using the training to teach the mapping function for the other three groups using the limited samples. The basic approach diagram is given below:

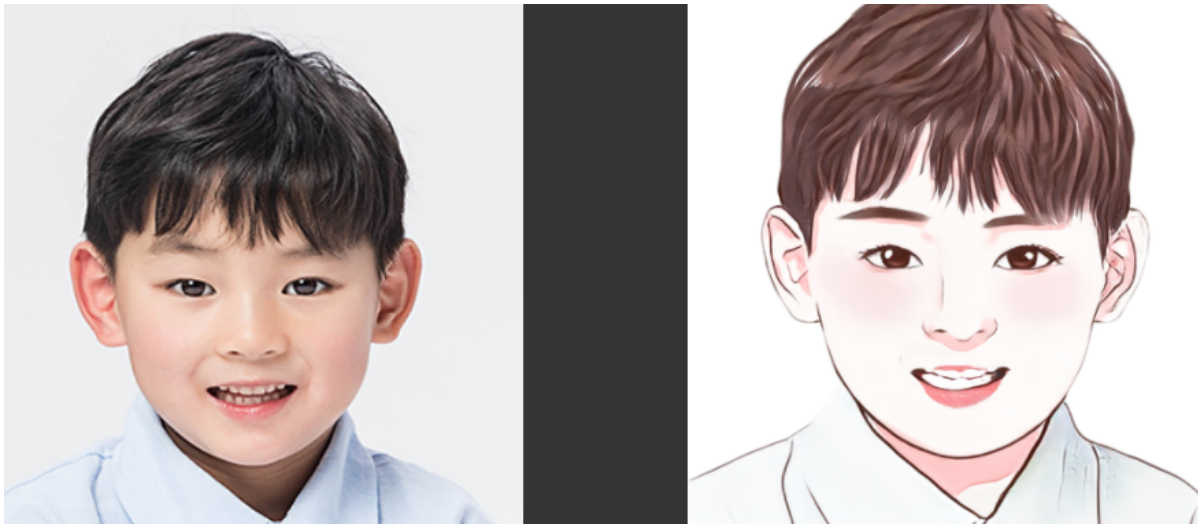


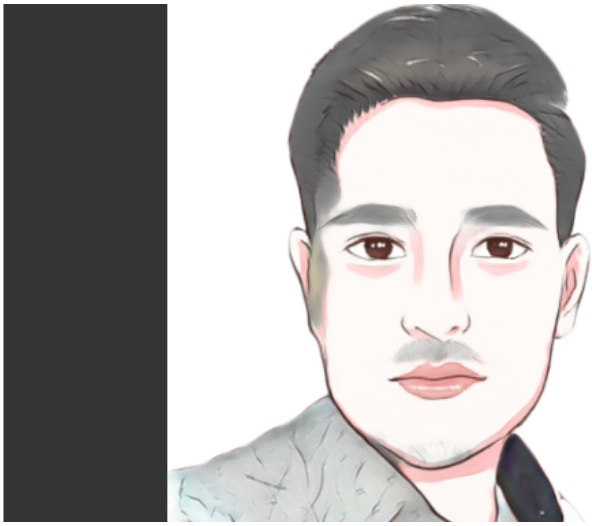
In knowledge transfer as can be seen we have the group-specific parameters and the shared parameters. The shared parameters are obtained by training on class 0 i.e. majority group. The group-specific parameters are obtained by training each group individually. It should be noted that selective backpropagation is used for training in groups 1,2,3 to avoid them from updating the shared parameters.

9. Result and Performance

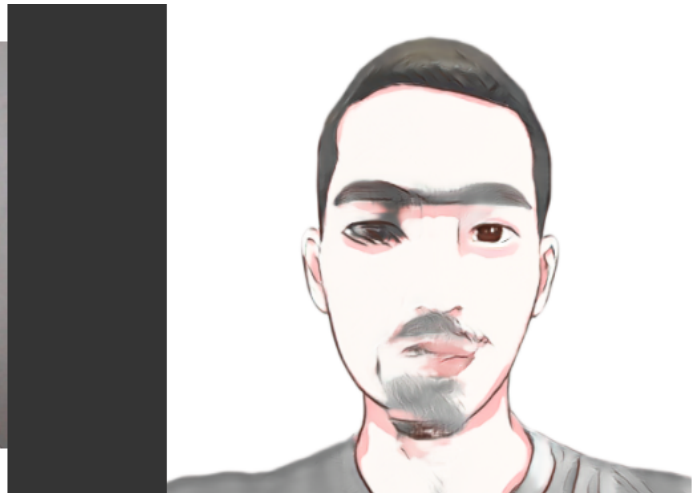
After the model is completely trained, we test it on some of the images shown below. We give a qualitative analysis of the results by showing some good results and some bad results.



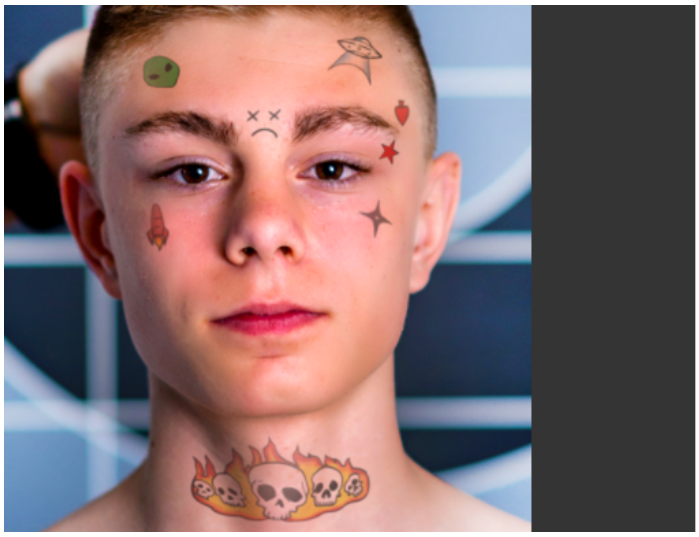


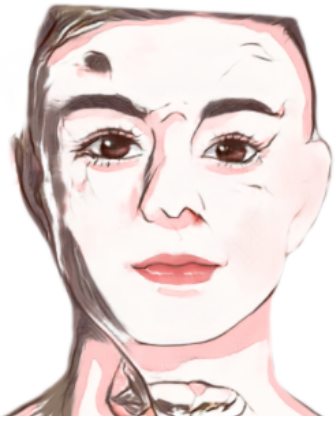


The above datasets provide decent results but, we can also see faulty results in the datasets shown below:

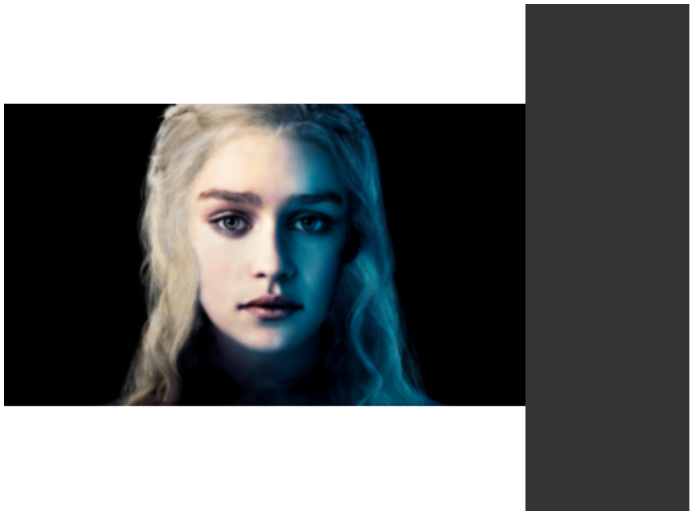


Cause for bad result: Facial hair and slightly shadowed image



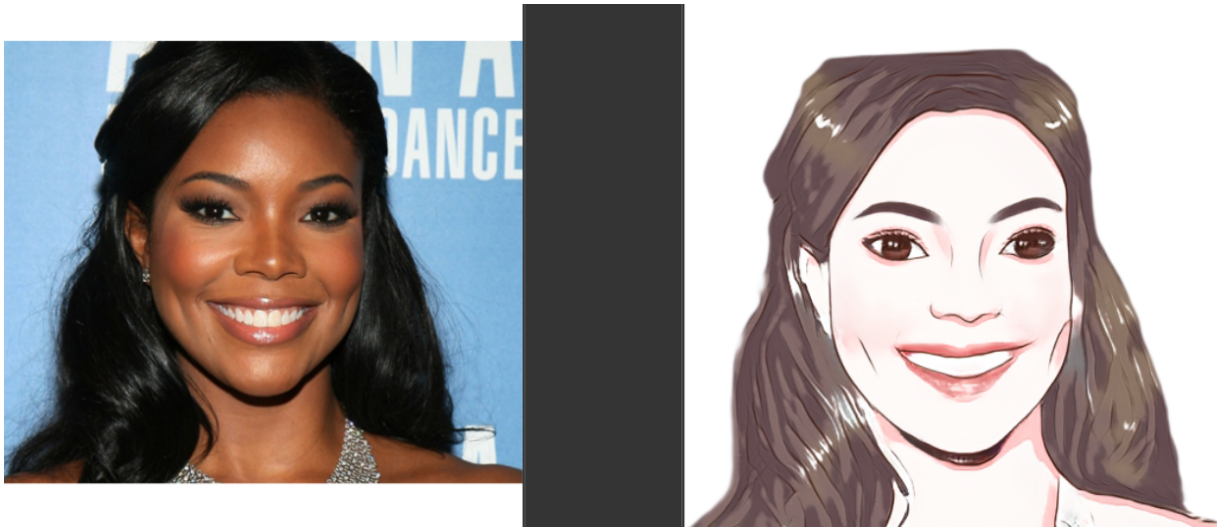


Cause for bad result: Tattoos not getting translated properly



Cause for bad result: shadowed and dark image

Cause for bad result: forehead not completely detected and obstructed by hair



In the above two cases, the complexion of the real image is not being portrayed in its corresponding cartoon image.

10. Conclusion and Further Work

As we can see from the results obtained above, the model is not perfect for images for groups other than group 0 because of the training process and methodology used by us. The images containing facial features like beards, mustaches (facial hair), other specifications like spectacles, tattoos, shadows on the face, etc are generating faulty results or slightly deformed images which can be a great improvement to the existing model. Also, the colour of the skin is not taken care of. For people with dark skin, the cartoon does not reflect similar skin colour. So, we plan to modify the model to train it on more datasets containing these parameters to improve the model. We also strive to create a model which does not rely on one specific group to train the other groups, as it is affecting our model to some extent.

References

1. https://arxiv.org/pdf/2007.13332.pdf
2. https://github.com/minivision-ai/photo2cartoon
3. https://arxiv.org/abs/1907.10830
4. https://arxiv.org/abs/1801.07698