# Investigative Study on Generative Models for Face2Sketch and Sketch2Face

1st Vaibhav Khandelwal, 35932110
*School of Computing and Communication)*
*MSc Data Science:SCC 413- Applied Data Mining*

***Abstract*—Image-to-image translation either from one colour to other or from one object picture as shown in Fig.2 is a computer vision and deep learning problem where our motive is to map input image features to the output image via training using either paired image or unpaired. It has its application in enforcing the law as well as in digital entertainment [39]. Different types of autoencoders and like sparse autoencoders, convolutional denoising autoencoders [38], various types of generative adversarial networks, etc have been developed in this field. Autoencoders are used to perform this task as it tries to form a connection between input and output images which has numerous applications such as data compression and data denosing. In this report we demonstrate the implication of different variations of autoencoders and Generative adversarial network (or GANs) on CUHK and Feret dataset to perform photo2sketch and sketch2photo usecase.**

## I. INTRODUCTION

Recently image-to-image translation has gained a huge hype and several applications has been developed so far as it has huge area of application like law has a huge implication of this technique especially when police has a sketch of criminal as in most of the cases the image of suspect isn't available with them and they need to find it in their database which becomes a hectic task [39] [42]. But often, the sketches and photos have huge differences thus making recognition of face sketch a daunting task. To match the images it is essential that both the images lies in the same modality, and if they aren't in the same modality then there can be two possible solution to this- One is to convert the image to sketch and then match both the images, secondly, by converting sketch into photo and match both the images [40], [41]. The sketch and photos are very different in-terms of texture and shape, but still artists have a great ability to present even the most minute characteristic of anyone's face and we can easily recognize people from sketch. From the Fig.1, it can be seen that both photos and sketch differs a lot due to being into different modalities and thus becomes hard in sketch recognition rather than photo recognition [42].

Our experiment of making a model to convert photo to sketch and sketch to photo using different methods of auto-encoders as shown in figure Fig.1 reveals the strength and weakness of different methods that has been deployed to carry our use-case. Additionally, different experimental setup are required for different methods based on the model parameters to be trained. Among the methods we considered, GANs variation is highly competitive to other auto-encoders.

There are various autoencoders but not all are appropriate for carrying out this use-case. We started with dense auto-encoders but results were vague. As convolution network works perfectly with images so for this use case we will make use of Convolutional autoencoders(CAE), Convolutional Variational autoencoders(CVAE), Pix2Pix Gans, and Cyclic GAN, and their results based on different optimization parameters will be mentioned and plotted at the end.

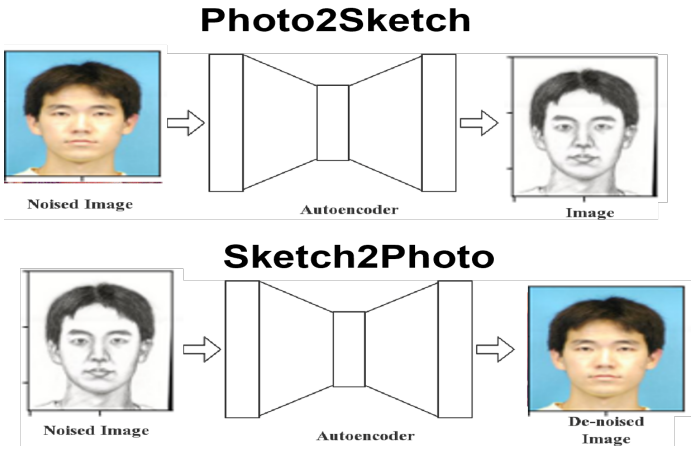


Fig. 1. Motive of the report

## II. LITERATURE REVIEW

Image generation and reconstruction analogy has had a long history in computer vision and deep learning world comprising plethora of successful experimentation in several different areas, including machine learning, deep learning, artificial intelligence, texture synthesis, and image-based rendering.

Image-to-Image translation is considered as pixel-to-pixel classification treating every pixel in an image to be independent from other pixels in the image [18, 19, 20]. But GANs consider the structural loss and various people have considered this loss type in their methods like SSIM metric[21], non-parametric loss [22]. In [32] the coloured to sketch image is carried out using computer vision gaussian blur filter.

Motivated by image synthesizing, Elfros proposed image quilting in 2001 [4], which is a used to synthesize a new image by putting together the small patches of existing image. In the same year Aaron Hertzmann developed a framework regarding image analogies [5] with which different types of filters could be produced over images.

Just like translation of one language to other like English to Spanish, the translation of image from one representation to other has been a major work carried out by different people over several years like the ones in Fig.2. Earlier each of these tasks were carried out explicitly via different people using techniques that could only be used particularly for one of the applications (eg,[6, 7, 8, 9, 12, 13, 14, 15, 16]).

In 2011 Andrew Ng introduced sparse and convolution autoencoders [10] to reconstructs the input image. As mentioned in [10], an autoencoder is a two-layer structured network that reconstructs the unlabelled input dataset given by $\{x_{i=1}^{(i)}{}^{m}\}$, in the motive to learn the $h(x^{(i)}; W, b) = \sigma(Wx^{(i)} + b)$ representation such that $\sigma(W^T h(x^{(i)}; W, b) + c)$ is approximately $x^{(i)}$,

$$\underset{W,b,c}{minimize} \sum_{i=1}^{m} ||\sigma(W^T \sigma(Wx^{(i)} + b) + c) - x^{(i)}||_2^2 \qquad (1)$$

[10] mentions the use of KL sparsity penalty while training sparse autoencoders, and based on their experimentation's claims that for large images convolution networked autoencoders should be used as sparse and densely connected ones does not not easily gets scaled.

Then in 2017 Isola gave an idea about Pix2Pix Generative Adversarial Network, or GAN, model [1] for Image-to-Image Translation with Conditional Adversarial Networks, and later in 2019 Silburt carried out identification of crater position and size using UNET architecture [3]. In [1] it has been explained on how pix2pix GAN generates an output of different modality than the input modality but based on the same input. So, for this it is required to pass a pair of images so that model learns to generate output image from its corresponding input image, and this model has various applications like conversion of satellite image to the google map, semantic segmented masked image like aerial image to street image, sketches to real images, black to white, day effects on photos to night effect etc as shown in Fig.2. Using this model several twitter users have posted their own artistic experiments [1].

In [3] they implemented UNET architecture taking moon surface images as training data and manually spotted crater images as target data as shown in Fig.3, and then used pixel-wise binary cross-entropy as a measure to calculate accuracy. With this they were able to achieve 92% recall.

Similar ideas of translating image-to-image have been applied by [33] to generate photographs from sketches or by [34] from attribute and semantic layouts. Recently, CoGAN by [35] shared a weight-sharing technique to find common features when doing unpaired image-to-image translation, and [36] extended this idea by combining various autoencoders.

Our goal in this report is to discuss the various methodologies to build a generative models for Face2Sketch and Sketch2Face. Although many significant steps have already been taken by people in this direction of which using convolution neural networks has been a breakthrough, but for our task it doesn't actually provide exact clear output because as per loss function used by CNN it aims to minimise the average


Fig. 2. Application of Pix2Pix GAN - Taken from original paper[1]

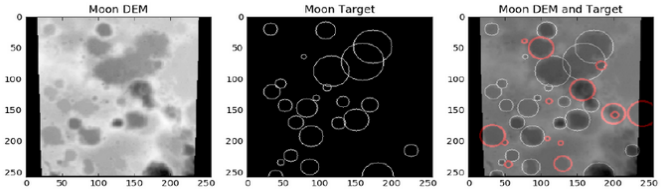


Fig. 3. Application of UNET to find craters - Taken from original paper[3]

euclidean distance between original image pixel and output image pixel, leading to blurry output [16, 17]. But GANs outperforms in this and provided sharp edge output. We would be using dense autoencoders, convolutional autoencoder [24], convolutional variational autoencoder [25, 26], pix2pix GAN [1], and cyclic GAN [23].

## III. PRELIMINARY ON METHODOLOGIES

### A. Dense Autoencoder

As it's very important to evaluate the performance of any model, so different metrics used in the classification modelling for measuring the model performance are:

### B. Convolutional Autoencoder(CAE)

Just like our normal autoencoder which consists of 3 parts-encoder, decoder and latent layer developed by fully connected layers. Similarly, in convolution autoencoder encoders and decoders are constructed using convolution layers and transposed convolution respectively. Both the input image size and output image size remains same. The sample CAE architecture can be seen in Fig.4, where all the blue layers form encoder and green layers form decoder.

***Network Architecture:*** :
The architecture that has been used in our use case consists of 6 convolution layers forming encoders and 6 transpose convolution layers to form decoder both having filter size

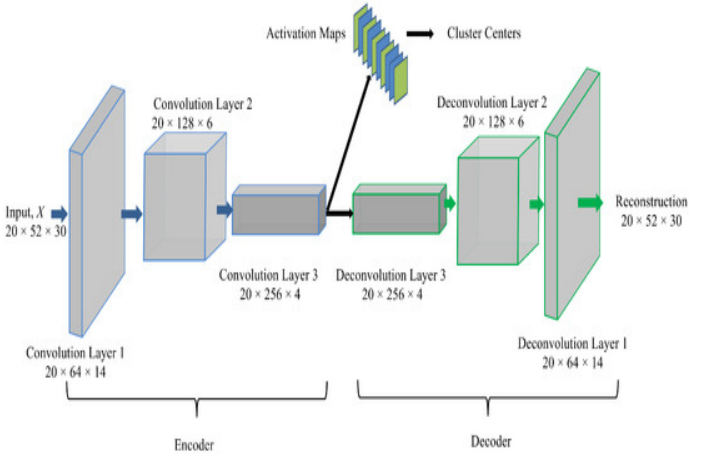


Fig. 4. Sample CNN autoencoder - Taken from [37]

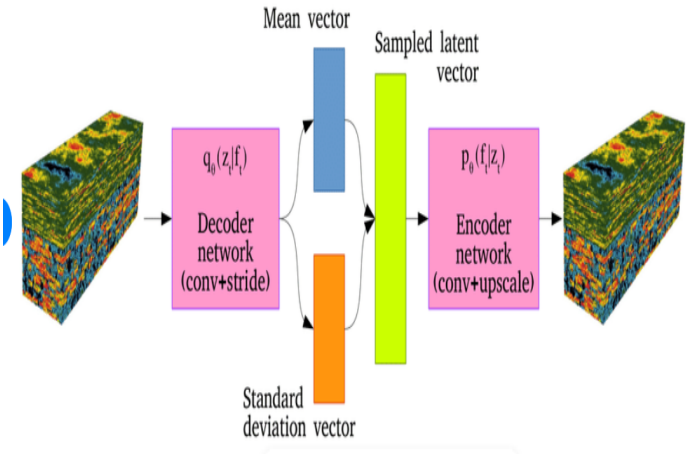


Fig. 5. Architecture of Convolutional Variationalautoencoder - Taken from [43]

of 4x4 and strides as 2. Except for the second layer in encoder which is followed by only leaky Relu, all other layers are followed by a combination of BatchNormalization and leaky relu. In decoder no BatchNormalization has been applied but all the layers are followed by a leky Relu. Input of size 256x256 is fed into the model as an input, and model also generates the images of same size which is then compared to the paired output image (like sketch image in case of photo2sketch). The output generated is compared to the original output image pixel by pixel and based on which cost function is minimised. The model is run at 500 epochs using distinct loss functions which are mean squared error, mean-absolute error and binary-cross entropy. The aim of the model is to learn to form the exact output by minimising the cost function. The model is compiled using adam optimizer and in each layer of convolution non-linear relu activation function was used. If sufficient computational resource is available then VGG16, Resnet, LeNet architecture could be used as encoder and decoder to get a great accuracy.

### *C. Convolutional Variational Autoencoder(CVAE)*

It is a combination of Convolutional autoencoder(CAE) and Variational autoencoder(VAE) where encoder is made up of CAE and decoder is created using VAE. The encoder architecture that has been used here is exactly similar to the one used in CAE and then at the output of encoder the features are flatten to be fed into the VAE. After this sampling occurs based in KL Divergence and then after passing through the decoder image is generated. The loss function here is a combination of KL divergence and mean-squared error. The architecture of CVAE is as shown in the figure Fig.5

### *D. Pix2Pix Generative Adversarial Networks*

Pix2Pix is a variation of conditional GANs but with slightly different architecture with a motive to do image-to-image translation. Here by applying the conditions on the given input target image is generated. In this model generator is made using U-NET architecture introduced in [2]. It takes in the original image and generates the segmented image. And the discriminator is created using "PatchGAN" classifier [27, 28].

***U-NET Architecture:*** :

In [2] the U-NET architecture was used for biomedical image segmentation to create high-resolution segmentation map and it's architecture is as shown in the figure Fig.6. In the figure multi-channel feature vectors are represented using blue block of 3 channels and convolution followed by non-linear relu activation function takes place between each block. Only valid part of the convolution is used that means 1 pixel border is lost every time a convolution using 3x3 feature map is done so that at the end large images could be broken down into individual tiles. Then max-pooling occurs which are shown via red arrows in Fig.6, and after each pooling the number of feature are doubled. As can be seen on the left part, we have four sets of two convolution layer with each set followed by a max-pooling with size 2x2 due to which the size of feature maps keeps on getting halved, and this is encoding part. After reaching to the base, decoding starts where up-convolution starts to take place. When going from base to the top concatenation happens which are shown using grey arrows coming from left towards right. During concatenation layers formed during decoding via up-convolution are concatenated with the ones formed during encoding at the same level, and we get segmented image at the end as our output.

The segmented image from generator along with the original image is passed into 70x70 discriminator which checks if the masked image from generator is the masked image of the original image, and results into binary output-discriminator loss and generator loss.

***Layered architecture of Generator and Discriminator:*** :

**Generator Encoder**:

C64-C128-C256-C512-C512-C512-C512-C512

Except first convolution layer with 64 filters i.e. C64, Batch-

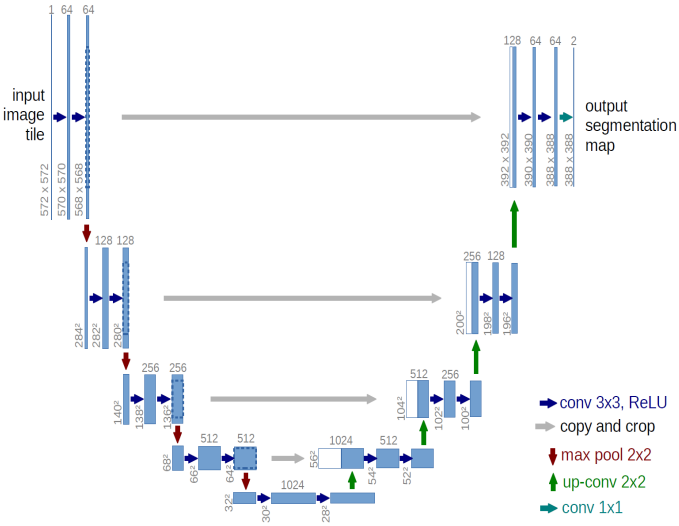


Fig. 6. Architecture of U-NET - Taken from Original Paper[2]

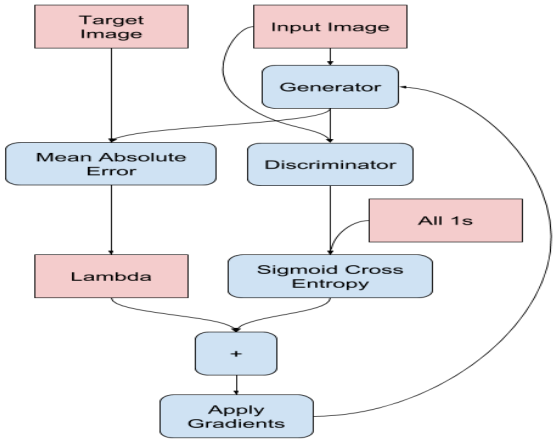


Fig. 7. Generator weights updating flow - Image from [31]

Normalization is applied across all the layers with all activation function being leakyRelu.

**Generator Decoder**:
C512-C512-C512-C512-C256-C128-C64
After the final C64 layer, convolution layer is applied to match the output channels followed by a tanh function. In decoder only normal Relu is applied in all the layers.

**Discriminator**:
C64-C128-C256-C512
Here all Relu's are leaky with slope 0.2 and BatchNormalization is not applied on the first C64 layer. After the end convolution, a comnvolution is applied to map the result to a one dimensional output.

The flow of the weights updation in generator can be seen from the Fig.7. Considering photo2sketch use case, our target images are the sketch images and the training are coloured images. The coloured images are passed through the generator and segmented images which is also known as fake image is given as output which is further compared to the target image using L1 loss (Mean Absolute error). Fake image also passes through the discriminator along with the original input image for training the discriminator and based on these two inputs binary-cross entropy is calculated. The L1 loss and binary cross-entropy loss are then combined together using lambda function which as per [1] is prescribed to take as 100 and gradient is obtained.

The discriminator is constructed following PathGAN architecture, where instead of classifying whole image as fake or real, each path of an image is classified. The discriminator convolves over whole image and returns the output by averaging all its responses. For our use case 70x70 patch size is used while training.

To optimize the network we have made use of the process mentioned in [29] according to which we to and fro one gradient-descent step between discriminator (D) and generator (G) that is we first do the D training and holding to it we move to train the G and like this it goes on which is what actually happens in GAN. Minibatch SGD along with Adam solver [20] is used while training the model. Instance normalization (i.e. keeping batch size as 1 when applying batch normalization) is used as it is found to be effective in image generation task as described by [30].

***Loss Function:*** :
Here two types of loss are taken into consideration- L1 loss from generator and binary cross-entropy from discriminator.

The objective loss function of a conditional GAN can be expressed as:

$$L_{cGAN}(G,D) = E_{x,y}[\log D(x,y)] + E_{x,z}[\log(1 - D(x, G(x,z)))] \quad (2)$$

Here G tries to minimize this objective function while D tries to maximize it. But as seen from the architecture we also need to consider the L1 loss which is given as:

$$L_{L1}(G) = E_{x,y,z}[||y - G(x,z)||_1] \quad (3)$$

So, combining both the losses we can get our required loss function given as:

$$G^* = \arg \min_G \max_D L_{cGAN}(G,D) + \lambda L_{L1}(G) \quad (4)$$

This method is preferred over prior because along with learning of input image to output image mapping also learns the loss function required to train it [1].

### *E. Cycle-Consistent Adversarial Networks*

Most of the time when we do image-to-image translation we don't have paired data. Looking at this issue Jun-Yan in 2020 created this architecture [23] of labelling two unordered image collection P and Q such that P can be translated to Q and Q can be translated to P. It has various applications like

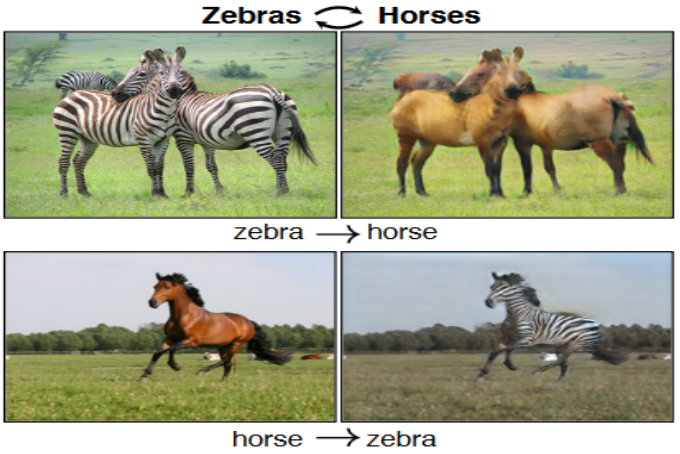


Fig. 8. Application of cycle GAN - Image from original paper [23]

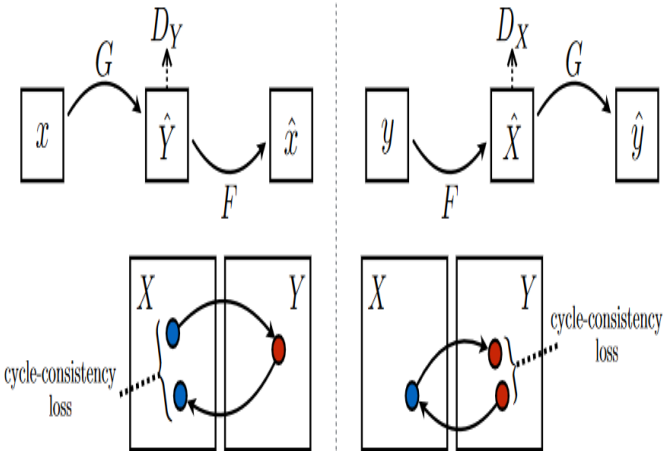


Fig. 10. Forward and backward cycle-consistency loss - Image from original paper [23]

inter-conversion of zebra to horse, monet to photos, summer photos to winter photos, etc as shown in Fig.8

here we actually try to map $\{x_i\}_{i=1}^n$ such that $x_i$ belongs to X and $\{y_i\}_{i=1}^n$ such that $y_i$ belongs to Y with no information of mapping between them. This method captures special characteristics from the input image and figures out a way to find a way to translate without having paired mapping.

Fig.9 shows two mapping functions G from X $\rightarrow$ Y and F from Y $\rightarrow$ X, and $D_x$ $D_y$ are its associated adversarial discriminators such that $D_Y$ encourages G to translate X into indistinguishable outputs with respect to Y, and vice versa for $D_X$ and F .

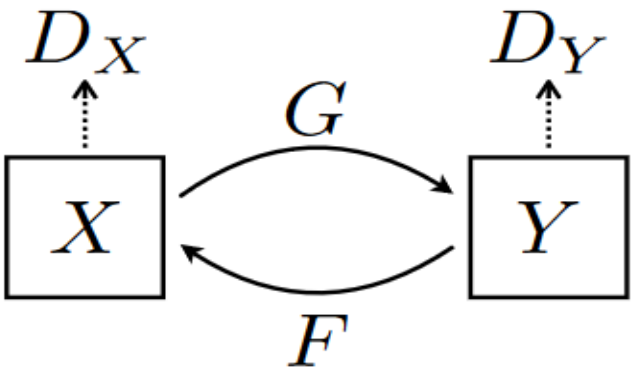


Fig. 9. Discriminator associated with two mapping function F and G - Image from original paper [23]

To regularize the mappings between X and Y, two cycle consistency losses is used to maintain the translation from X to Y and Y to X which is the starting image as shown in Fig.10.

***Network Architecture:*** :
Here 70x70 patch GANs discriminator is constructed. Here discriminator weights are updated using 50 previously created buffered images from the history to reduce model oscillation and Adam solver with stride 1 has been used.

Considering C7S1-K denote a 7×7 Convolutionlayer-InstanceNormalization- ReLUactivation layer with K filters and stride **1**. dk denotes a 3 × 3 Convolutionlayer-InstanceNormalization-ReLUactivation with K filters and stride **2**. A residual block Rk that contains two 3×3 convolutional layers with the same number of filters on both layers. Uk denotes a 3×3 fractional-strided-Convolutionlayer-InstanceNormalization-ReLUactivation layer with K filters and stride 1/2.

The network with 9 residual blocks consists of:
C7S1-64,D128,D256,R256,R256,R256,R256,R256,
R256,R256,R256,R256,U128U64,C7S1-3

Its has a cons that it is not able to provide back the exact same color gradient.

## IV. EXPERIMENTAL RESULTS

### A. Experimental Setup

All the methods that have been mentioned here in this report were implemented on CUHK and Feret dataset. CUHK contains 88 images of each coloured images and it's corresponding sketch images, and in Feret 745 such images of each are available.

We used Google colab for carrying out all the modelling, having specification- n1-highmem-2 instance, 2vCPU @ 2.2GHz, 13GB RAM, 78GB Free Disk Space, 64GB disk space GPU instance, 90 minutes idle cut-off time with maximum upto 12 hours as shown in Fig.11

The programming language used for modelling is python, libraries used for plotting images is opencv and matplotlib, and for modelling keras **2.7.0** (backend with Tensorflow) was used.

As the image in dataset was of very large size so it became impossible to make a very deep network as if image of size 256 x 256 was taken as input, it was leading to the crashing of session due to very large trainable parameters.

```
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 63
model name      : Intel(R) Xeon(R) CPU @ 2.30GHz
stepping        : 0
microcode       : 0x1
cpu MHz         : 2299.998
cache size      : 46080 KB
physical id     : 0
siblings        : 2
core id         : 0
cpu cores       : 1
apicid          : 0
initial apicid  : 0
fpu             : yes
fpu_exception   : yes
cpuid level     : 13
```

Fig. 11. Specification of Google Colab used for experimentation - Image from Author

Also due to the limitation on GPU usage per day we couldn't go for large epochs for training. So all the experimentation was performed with limited epochs but varying optimization parameters keeping computational resources capacity in mind.

Due to availability of low computational resource, CUHK dataset has been used as it's a small dataset and can easily run on our machine without crash. As image size is large and have few unnecessary part so we cropped the image using numpy and resized into 256x256 pixels. Then after normalization we splitted the data into train and test.

The loss functions used for comparing different models are mean-squared error(MSE), mean-absolute error(MAE) and Binary-cross entropy.

### *B. Training Methods and Testing results*

*1)* ***Convolutional Autoencoder:*** :

**For Pic2Sketch** :

The sketch in Fig.12 are obtained when convolutional autoencoder was applied over coloured images at 500 epochs and using three different loss functions- mean-squared error, mean-absolute error and binary-cross entropy. It was observed that sketches obtained after applying mean-squared error loss functions are quite better than mean-absolute error, but when using binary-cross entropy the result is worst. The graphs in Fig.12 are the validation loss and training loss over epochs, and it can be observed that both the training and validation loss is decreasing at a good rate in case of mean-squared error as compared to other two. Thus, in the case of photo2sketch, using mean-squared error loss function will give a good results.

**For Sketch2Pic**:

When this convolutional autoencoder was ran for sketch2photo modelling results obtained were very similar to that of photo2sketch. Here also, for model ran by taking mean-squared error loss proved out to be better than other two as can be seen from Fig.14. But in this case model with binary-cross functions seems to outperform mean-absolute error as can be seen from Fig.14 and Fig.15

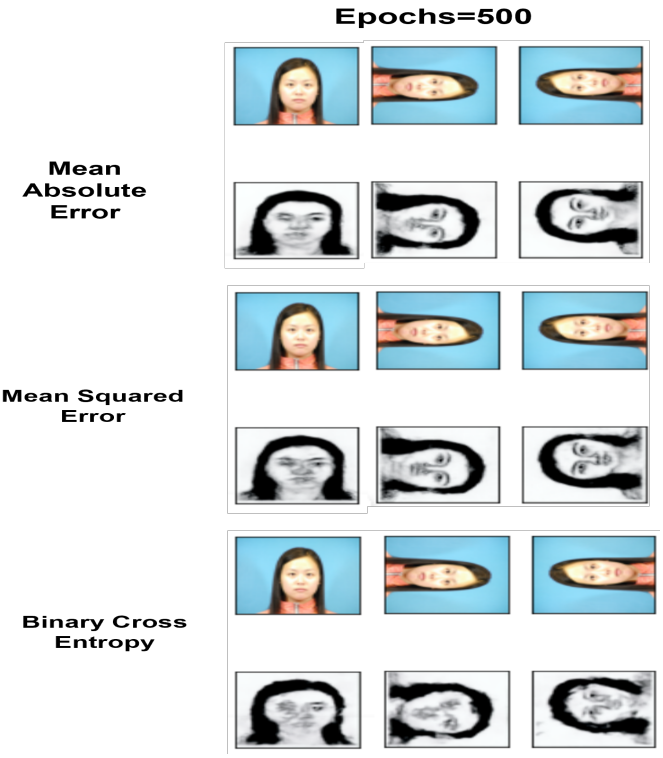


Fig. 12. CNN autoencoder outputs on coloured test images using different loss functions- MSE, MAE, BCE

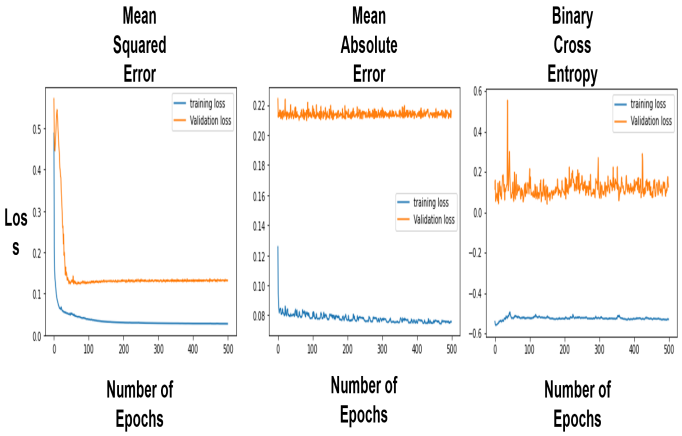


Fig. 13. CNN autoencoder Photo2Sketch training and validation loss on different epochs and different loss functions- MSE, MAE, BCE

In Fig.15 we can see that colour is not completely filled in case of mean-absolute error but colour is completely filled but is blurred in case of binary-cross entropy. In case of mean-squared error image is blurred but a bit less than other two models. Thus, in the case of sketch2photo, using mean-squared error loss function will give a good results.

*2)* ***Convolutional Variational Autoencoder:*** :

For both the use case model ran three times at different latent layer size- 2, 4, and 12 and the result obtained is as

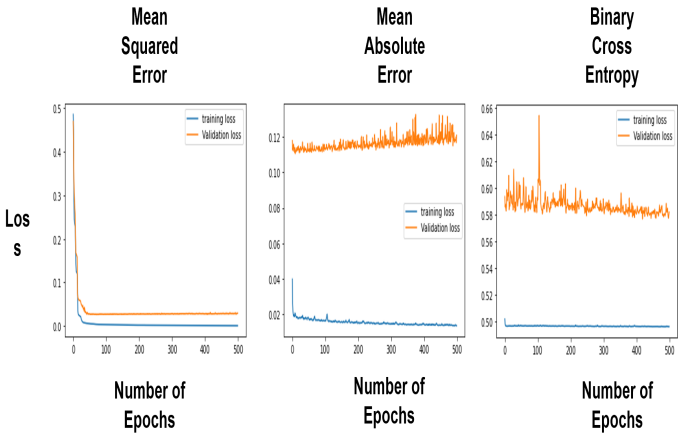


Fig. 14. CNN autoencoder Sketch2Photo training and validation loss on different epochs and different loss functions- MSE, MAE, BCE

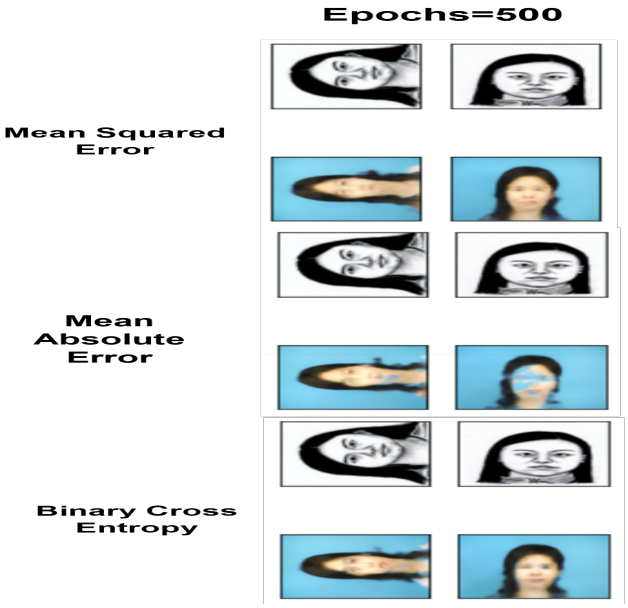


Fig. 15. CNN autoencoder outputs on sketch test images using different loss functions- MSE, MAE, BCE

shown in Fig.16 and Fig.19 . From this figure it can be seen that the one at the bottom is more clear because it was ran for latent layer size 12 and top one with size 2.

Also, from the Fig.17 and Fig.18 it can be seen for both the use cases C is the better model with decreasing validation loss as compared to other where with each epoch loss is increasing.

*3) **Pix2Pix GANs:*** :

**For Pic2Sketch** Following sketch are observed when ran the model for 20, 40 and 100 epochs respectively as shown in Fig.20. Due to GPU limitation couldn't run for more epochs and on larger dataset due to which not getting much accuracy. But with the increase in dataset and epochs this model gives the best result. But as can be seen as more epochs are taken sketch of the photos started to become more clear.

As number of epochs increased the generator error also

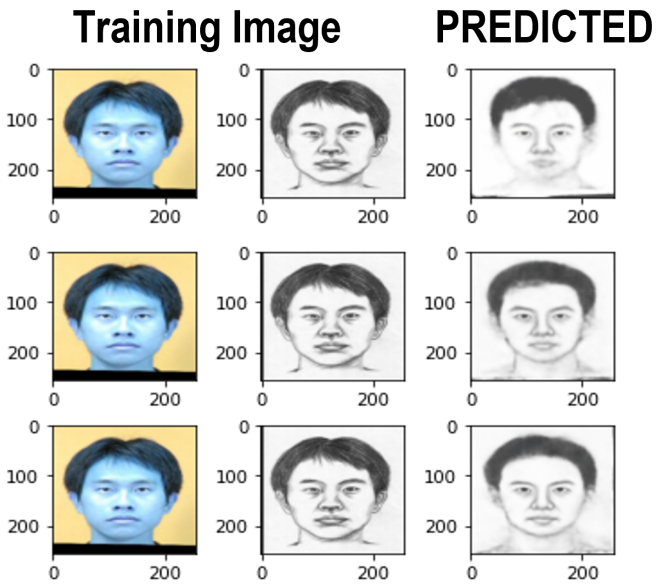


Fig. 16. CVAE autoencoder outputs on coloured test images using different latent feature size value- Top one being 2 and bottom one being size 12

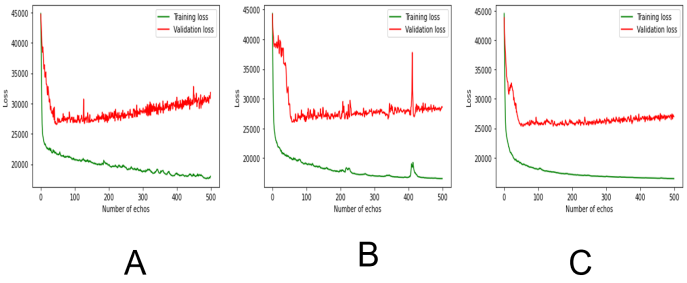


Fig. 17. Training and Validation loss of Photo2Sketch on different latent layer size- A:2, B:4, C:12 size

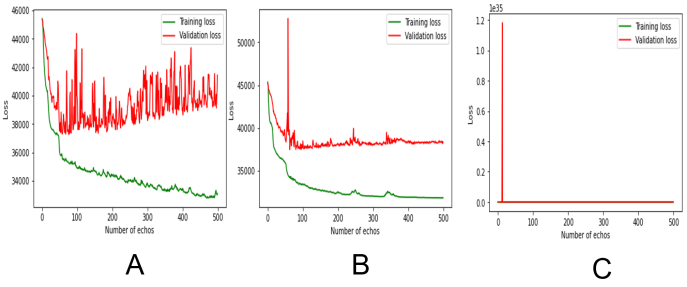


Fig. 18. Training and Validation loss of Sketch2Photo on different latent layer size- A:2, B:4, C:12 size

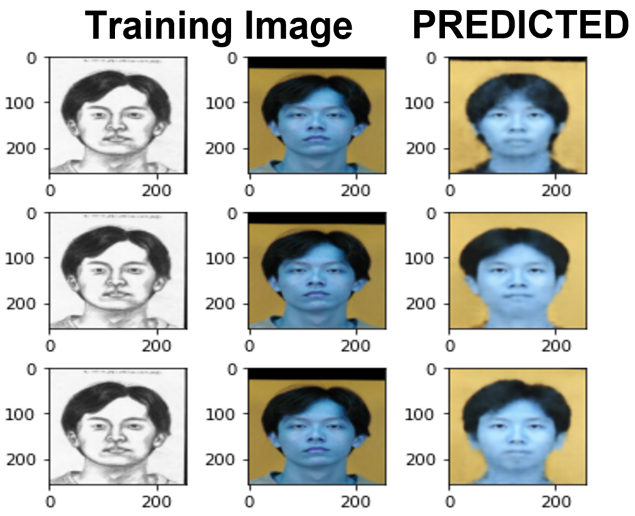


Fig. 19. CVAE autoencoder outputs on sketch test images using different latent feature size value- Top one being 2 and bottom one being size 12

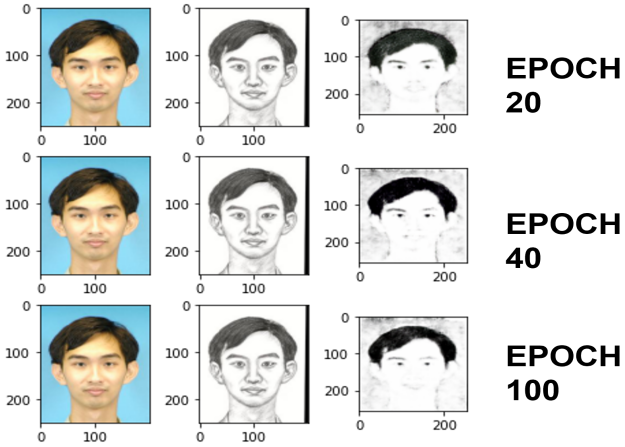


Fig. 20. Prediction of the model over various epochs

started to decrease as can be seen from Fig.21 where discriminator and generator loss over various epochs has been plotted.

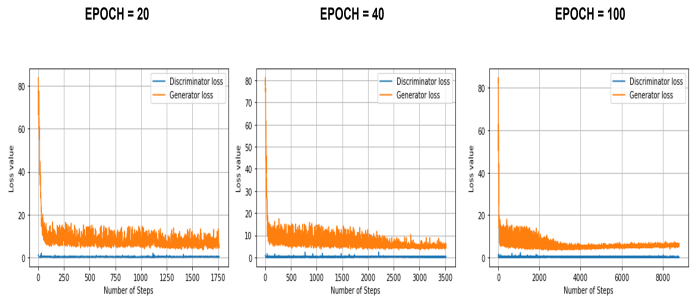


Fig. 21. Discriminator and Generator loss over various epochs

**For Sketch2Pic** With every increase in epoch this model performs well on sketch2photo as can be seen from the Fig.22 and the generator loss decreases with increase in epochs and shows the same behaviour as that of photo2sketch Fig.21.

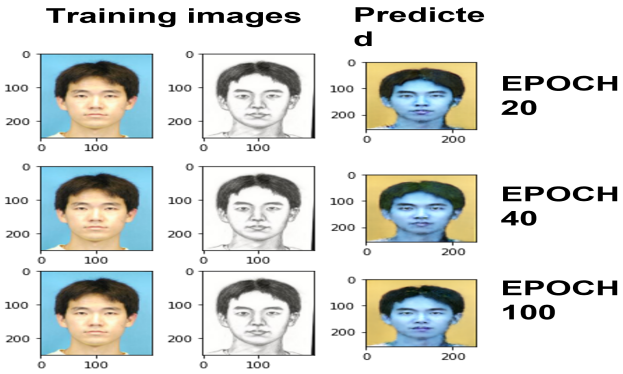


Fig. 22. Prediction of the model over various epochs - Image from Author

*4) **Cycle-Consistent Adversarial Networks**:* : As in cyclic-GAN both photo2sketch and sketch2photo runs together so we found our output using one model itself that means model was trained using photo to sketch but to the speciality of this model it also learnt the sketch to photo and following results as shown in Fig.23 were observed. From the figure it can be seen only colour is not proper but face is almost coming accurate in both the cases.

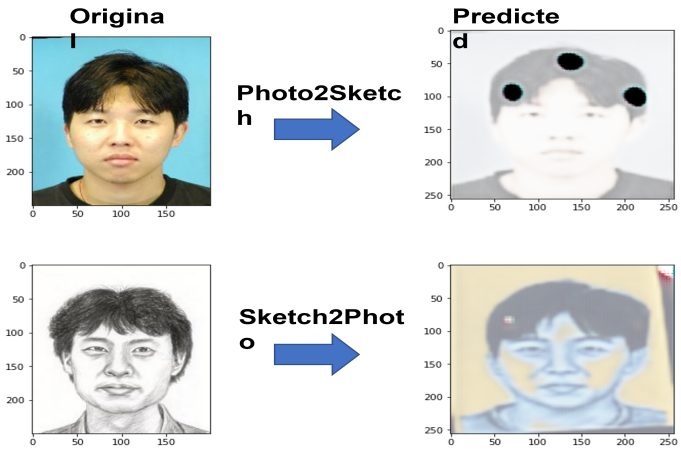


Fig. 23. Prediction of the cycle GAN model - Image from Author

Also, while training the model, the generator loss and discriminator loss of both A and B in A-¿B model were found to be decreasing as shown in Fig.24.

## V. DISCUSSION

Depending on the use-case different types of image-to-image translation technique works. Started with Dense autoencoder but results were not that promising, so switched to convolution as convolution networks are specifically for image processing which gave a good result when applied convolutional autoencoder. But as it varies in terms of distribution when tested with some new image. So to overcome this Variational autoencoder was used as it takes into consideration

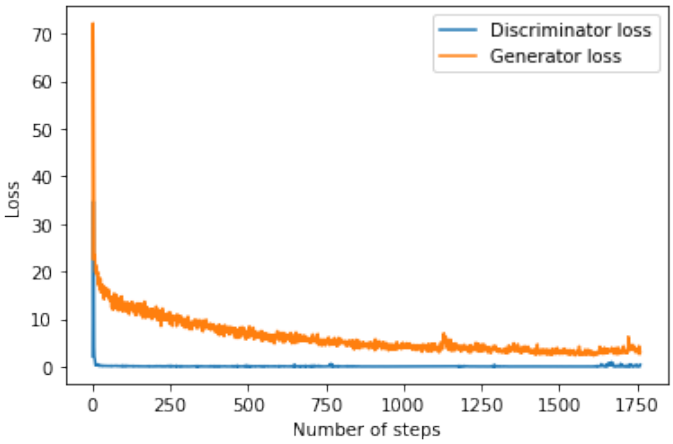


Fig. 24. Discriminator and Generator loss over steps - Image from Author

the spatial distribution. But GANs actually outperforms all especially Cyclic GAN which even with very low epochs gave a good result. So based on computation resources selection between autoencoders and GANs should be made.

## VI. CONCLUSION

In this paper we proposed encoders and GAN based model for photo2sketch and sketch2photo usecases. It has potential applications in various areas like law enforcement and other photo search application. If computational resources is not a limitation then cyclic and pix2pix GAN will outperform all the models but if it does then Convolutional Variational autoencoders can give a good result.

The codes of all the four methods used are available in the following location:
https://github.com/vaibhavcodes/SCC-413_Assignment

## REFERENCES


[1] Isola, P., Zhu, J.Y., Zhou, T. and Efros, A.A., 2017. Image-to-image translation with conditional adversarial networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1125-1134).

[2] Ronneberger, O., Fischer, P. and Brox, T., 2015, October. U-net: Convolutional networks for biomedical image segmentation. In International Conference on Medical image computing and computer-assisted intervention (pp. 234-241). Springer, Cham.

[3] Silburt, A., Ali-Dib, M., Zhu, C., Jackson, A., Valencia, D., Kissin, Y., Tamayo, D. and Menou, K., 2019. Lunar crater identification via deep learning. Icarus, 317, pp.27-38.

[4] Efros, A.A. and Freeman, W.T., 2001, August. Image quilting for texture synthesis and transfer. In Proceedings of the 28th annual conference on Computer graphics and interactive techniques (pp. 341-346).

[5] Hertzmann, A., Jacobs, C.E., Oliver, N., Curless, B. and Salesin, D.H., 2001, August. Image analogies. In Proceedings of the 28th annual conference on Computer graphics and interactive techniques (pp. 327-340).

[6] Fergus, R., Singh, B., Hertzmann, A., Roweis, S.T. and Freeman, W.T., 2006. Removing camera shake from a single photograph. In ACM SIGGRAPH 2006 Papers (pp. 787-794).

[7] Buades, A., Coll, B. and Morel, J.M., 2005, June. A non-local algorithm for image denoising. In 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) (Vol. 2, pp. 60-65). IEEE.

[8] Chen, T., Cheng, M.M., Tan, P., Shamir, A. and Hu, S.M., 2009. Sketch2photo: Internet image montage. ACM transactions on graphics (TOG), 28(5), pp.1-10.

[9] Shih, Y., Paris, S., Durand, F. and Freeman, W.T., 2013. Data-driven hallucination of different times of day from a single outdoor photo. ACM Transactions on Graphics (TOG), 32(6), pp.1-11.

[10] Le, Q.V., Ngiam, J., Coates, A., Lahiri, A., Prochnow, B. and Ng, A.Y., 2011, January. On optimization methods for deep learning. In ICML.

[11] Chu, C.T., Kim, S., Lin, Y.A., Yu, Y., Bradski, G., Olukotun, K. and Ng, A., 2006. Map-reduce for machine learning on multicore. Advances in neural information processing systems, 19.

[12] Laffont, P.Y., Ren, Z., Tao, X., Qian, C. and Hays, J., 2014. Transient attributes for high-level understanding and editing of outdoor scenes. ACM Transactions on graphics (TOG), 33(4), pp.1-11.

[13] Long, J., Shelhamer, E. and Darrell, T., 2015. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3431-3440).

[14] Eigen, D. and Fergus, R., 2015. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In Proceedings of the IEEE international conference on computer vision (pp. 2650-2658).

[15] Yoo, D., Kim, N., Park, S., Paek, A.S. and Kweon, I.S., 2016, October. Pixel-level domain transfer. In European conference on computer vision (pp. 517-532). Springer, Cham.

[16] Zhang, R., Isola, P. and Efros, A.A., 2016, October. Colorful image colorization. In European conference on computer vision (pp. 649-666). Springer, Cham.

[17] Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T. and Efros, A.A., 2016. Context encoders: Feature learning by inpainting. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2536-2544).

[18] Iizuka, S., Simo-Serra, E. and Ishikawa, H., 2016. Let there be color! Joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification. ACM Transactions on Graphics (ToG), 35(4), pp.1-11.

[19] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y., 2014. Generative adversarial nets. Advances in neural information processing systems, 27.

[20] Larsson, G., Maire, M. and Shakhnarovich, G., 2016, October. Learning representations for automatic colorization. In European conference on computer vision (pp. 577-593). Springer, Cham.

[21] Wang, Z., Bovik, A.C., Sheikh, H.R. and Simoncelli, E.P., 2004. Image quality assessment: from error visibility to structural similarity. IEEE transactions on image processing, 13(4), pp.600-612.

[22] Li, C. and Wand, M., 2016. Combining markov random fields and convolutional neural networks for image synthesis. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2479-2486).

[23] Zhu, J.Y., Park, T., Isola, P. and Efros, A.A., 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the IEEE international conference on computer vision (pp. 2223-2232).

[24] Mao, X.J., Shen, C. and Yang, Y.B., 2016. Image restoration using convolutional auto-encoders with symmetric skip connections. arXiv preprint arXiv:1606.08921.

[25] Kingma, D.P. and Welling, M., 2019. An introduction to variational autoencoders. arXiv preprint arXiv:1906.02691.

[26] Simonovsky, M. and Komodakis, N., 2018, October. Graphvae: Towards generation of small graphs using variational autoencoders. In International conference on artificial neural networks (pp. 412-422). Springer, Cham.

[27] Mirza, M. and Osindero, S., 2014. Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784.

[28] Demir, U. and Unal, G., 2018. Patch-based image inpainting with generative adversarial networks. arXiv preprint arXiv:1803.07422.

[29] Hinton, G.E. and Salakhutdinov, R.R., 2006. Reducing the dimensionality of data with neural networks. science, 313(5786), pp.504-507.

[30] Xie, S., Huang, X. and Tu, Z., 2016, October. Top-down learning for structured labeling with convolutional pseudoprior. In European Conference on Computer Vision (pp. 302-317). Springer, Cham.

[31] Tensorflow pix2pix https://www.tensorflow.org/tutorials/generative/pix2pix

[32] Khayan, A. and Khoenkaw, P., 2021, March. Automatic Pencil Sketch Landscape Image Generation From Photograph. In 2021 Joint International Conference on Digital Arts, Media and Technology with ECTI

Northern Section Conference on Electrical, Electronics, Computer and Telecommunication Engineering (pp. 27-30). IEEE.
[33] Karacan, L., Akata, Z., Erdem, A. and Erdem, E., 2016. Learning to generate images of outdoor scenes from attributes and semantic layouts. arXiv preprint arXiv:1612.00215.
[34] Tyleček, R. and Šára, R., 2013, September. Spatial pattern templates for recognition of objects with regular structure. In German conference on pattern recognition (pp. 364-374). Springer, Berlin, Heidelberg.
[35] Liu, M.Y. and Tuzel, O., 2016. Coupled generative adversarial networks. Advances in neural information processing systems, 29.
[36] Liu, M.Y., Breuel, T. and Kautz, J., 2017. Unsupervised image-to-image translation networks. Advances in neural information processing systems, 30.
[37] Chadha, G.S., Islam, I., Schwung, A. and Ding, S.X., 2021. Deep Convolutional Clustering-Based Time Series Anomaly Detection. Sensors, 21(16), p.5488.
[38] Gondara, L., 2016, December. Medical image denoising using convolutional denoising autoencoders. In 2016 IEEE 16th international conference on data mining workshops (ICDMW) (pp. 241-246). IEEE.
[39] Wang, X. and Tang, X., 2008. Face photo-sketch synthesis and recognition. IEEE transactions on pattern analysis and machine intelligence, 31(11), pp.1955-1967.
[40] Koshimizu, H., Tominaga, M., Fujiwara, T. and Murakami, K., 1999, October. On KANSEI facial image processing for computerized facial caricaturing system PICASSO. In IEEE SMC'99 Conference Proceedings. 1999 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 99CH37028) (Vol. 6, pp. 294-299). IEEE.
[41] Iwashita, S., Takeda, Y. and Onisawa, T., 1999, August. Expressive facial caricature drawing. In FUZZ-IEEE'99. 1999 IEEE International Fuzzy Systems. Conference Proceedings (Cat. No. 99CH36315) (Vol. 3, pp. 1597-1602). IEEE.
[42] Liu, Q., Tang, X., Jin, H., Lu, H. and Ma, S., 2005, June. A nonlinear approach for face sketch synthesis and recognition. In 2005 IEEE Computer Society conference on computer vision and pattern recognition (CVPR'05) (Vol. 1, pp. 1005-1010). IEEE.
[43] Temirchev, P., Simonov, M., Kostoev, R., Burnaev, E., Oseledets, I., Akhmetov, A., Margarit, A., Sitnikov, A. and Koroteev, D., 2020. Deep neural networks predicting oil movement in a development unit. Journal of Petroleum Science and Engineering, 184, p.106513.