Image Colorization Using Generative Adversarial Networks

Ritwik Sinha
School of Electronic Engineering and Computer Science (EECS)
Queen Mary University of London
London, UK
ec211251@qmul.ac.uk
Project Supervisor: Sebastian Berns

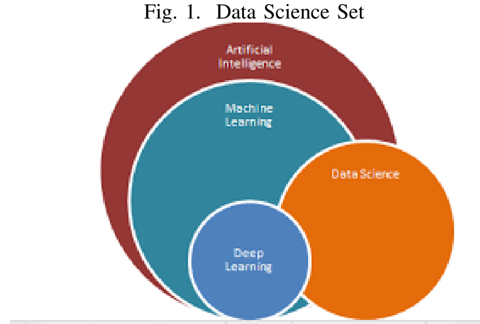
***Abstract*—Data is the new gold, and it is increasing at humongous rate. The data could be structured, semi-structured or unstructured. Out of these three types of data, image data comes under unstructured data. Since, this project revolves completely around image data, we can say, the type of data used in this project is unstructured. Now, as we will use images, images could be either gray scaled or colored. Gray scaled images has 1 channel while colored or RGB scaled images has 3 channels (Red, Green and Blue Channels). For a user to understand the characteristics of image, we should prefer colored image data. Image colorization is done in order to visualise images better as it is quite difficult in the case of gray scale images. This project comprises of implementation of Generative Adversarial Networks in order to perform image colorization for gray scale images. The architecture of the model implemented is divided into two parts, first is the generator, it is responsible for generating colored images, for this project the generator takes coloured image as an input, convert it into gray scale image and further generates a colored image. This is done in order to fool the discriminator while predicting the actual colored and generated colored images. The colored image coming from generator is basically an artificially generated colored image. The second part of the architecture is the discriminator, which helps to classify the input coloured image and the coloured image obtained from GAN. The second part solves a problem that, it will ultimately help a user to be on a safer side of reality and does not get misguided due to artificial images present on the internet. There are enormous unsecured websites which can misguide a user claiming the product as real with the help of artificially generated images. The discriminator part differentiates between the original colored image from generator's colored images and can eventually save user from unsecured sources.**

***Index Terms*—Image processing, Architecture, Hypercolumn, GAN, Grayscale Images, RGB Images, Generator, Discriminator, CNN**

I. Introduction

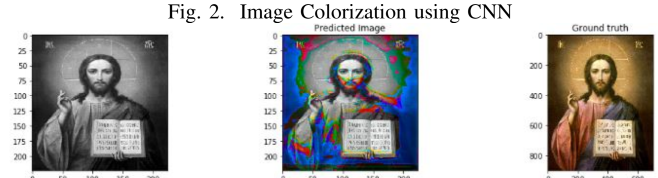
Colorization of images is an integral part of image processing in the world of data science. Figure 1 illustrates that Artificial intelligence is the main set of data science under which Machine Learning and Deep Learning comes as the subset of it. However, AI is just a part of Data Science. As Data Science includes various other things, such as Databases, Data Visualisations, Mathematical Modeling etc. Coming back to the image processing, it comes under deep learning set of Artificial Intelligence. Currently, under image processing, enormous study is under supervision related to face detection and recognition. In this century the biggest threat to human society is human itself, hence AI is regularly providing the safety to the humans with the help of image processing. Our topic of interest which is image colorization is other aspect of image processing.

Fig. 1. Data Science Set



In order to perform image colorization, the main incentive is to bring the liveliness of old era of black and white images to the current era of colored images. Image colorization not only helps to glorify old images but it also helps to provide a way to broaden the scope of identifying old objects and sculptures. Apart from implementing Generative Adversarial Networks, there were certain studies related to image colorization has happened with the help of Convolutional Neural Network for colorizing old black and white image of Jesus Christ. In figure 2, we get a comparative view of black and white image of Jesus Christ, a colorized image output of CNN, and the original colored image of Jesus Christ. The expected output of our project is better than the output of this Convolutional Neural Network, as we are implementing Generative Adversarial Networks which is very effective in performing image colorization.

Fig. 2. Image Colorization using CNN



With the introduction of deep learning, the problem of image colorization has been solved in a more easier manner. The old age gray images of which we do not have color scale, deep learning helps to provide color into it. However, there are many deep learning algorithms available and choosing the correct deep learning algorithm for our generator and discriminator is a task to do. There are four approaches to colorize an image, one is to use Convolutional Neural Networks (CNN) for both parts of architecture, second is to using Generative Adverserial Networks (GAN) for both parts of architecture, the other two is using either of the network for either of the parts of architecture. CNN's are capable of extracting useful features out of the images automatically. On the other hand, GAN's are capable of generating images, restoring facial images and make an image clear if there exists blurriness into it. One of the drawback of CNN is that, it takes long time to train in comparison to GAN because in CNN, the intermediate layers contains useful information, and the classification is not done only with the output of the last layer, thus training occurs through every layer of CNN in each epoch. There are other advantages and disadvantages of both networks, but for the scope of this project, the above mentioned features of both networks simplifies the problem of choosing the best network for the architecture. Hence, for our project, we will be using GAN, as this part will be responsible for generating images, colorizing it and classifying the images as original image or generated image (real or fake).

There are different types of GAN's, out of them we will be implementing CD-GAN (Cyclic Discriminative GAN) [17]. CD-GAN generates high quality images, gives better validation accuracy and is much faster than the auto encoder which is used in CNN, as CD-GAN does not require data pre-processing. Figure 3 shows a sample case of how does a CD-GAN works.

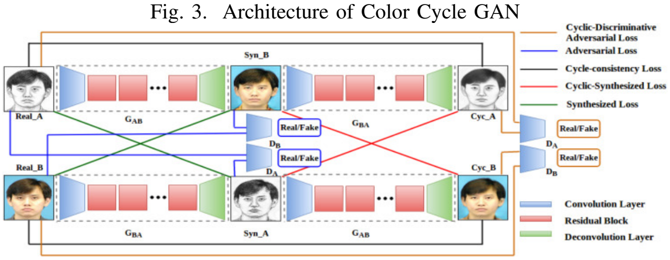


Fig. 3. Architecture of Color Cycle GAN

II. Related Work

With the expansion of machine learning and introduction of deep learning a lot of experimentation has occurred in the field of image colorization. There are two ways of performing image colorization, one is user guided approach and another is automatic approach [6]. In user guided approach, it is the responsibility of a user to provide information about the colors present in an image with the help of a reference image. Every part of an image which needs to be colored is provided with a specific color point, that particular color is spread in a smaller region close to it. However, this approach is not very much appreciated as it is up to the understanding of the user about the color codes of an image and choosing of reference image. In order to overcome this issue, we generally prefer automatic approach. In this approach there is no specific color point which is provided, but it requires some useful information present in an image, and this process is known as feature extraction. Deep learning is more than capable of extracting meaningful information from the images and learn the color codes. Since, in the real world, often we do not have enough data to train a model and get the desired results, thus in order to run deep learning algorithm, certain studies follow the usage of pre-trained models which were already deployed on millions of data points. These pre-trained model not only helps to make the model robust but also it helps to fasten the learning rate. Although, there exists a drawback of using pre-trained models. The major drawback of using them is data mismatch. It is not in every case that we get the pre-trained models which were trained in similar type of data or images in case of image processing as what would be required for our particular model. The color scheme or codes used in the data while training the pre-trained model might be drastically different to the data of our concern.

In order to understand the drawback of using pre-trained model, let us assume most of the images of sky used in pre-trained model are blue in color, but in our data the images of sky is mostly taken during sunrise and sunset, thus, the color code predicted by pre-trained model for our image would basically be blue in color. Since, blue color for sky is correct but for our image colorization process, it is incorrect because during sunrise or sunset the sky appears to be yellow or orange or red in color, and we expect the same from the model predictions.

Moving onto the methods deployed in the past for manual image colorization, there are certain methods commonly known as luminance keying, color transfer method, unsupervised method of coloring images using an example and iterative probabilistic relaxation method [4]. In figure 4 we can see the working of Luminance keying method. The girl standing on the left in the picture is the reference image and the boy standing on the right is the image which gets colorized with this method.

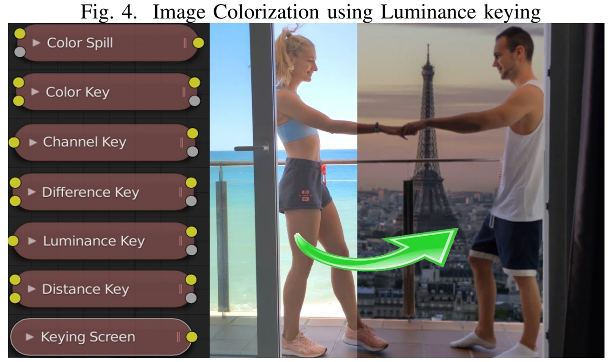
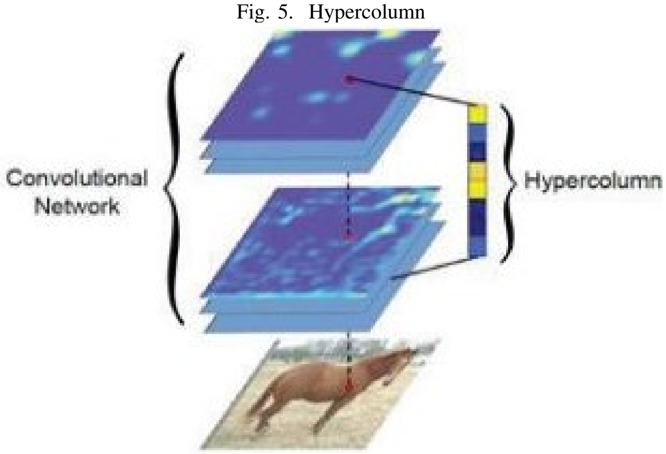


Fig. 4. Image Colorization using Luminance keying

Luminance keying and color transfer methods are interrelated to each other as the later overcome a drawback of the prior. In luminance keying method, a function is used to map each level of luminance of image to a color space. Since, it is non-viable to wrap whole color space, thus, color transfer method is used thereafter. Under this method, a reference color image is provided using which the gray image tends to get colored. This method extracts luminance information and color textures from reference image and colorizes the gray image. The unsupervised method is also similar to the previous methods, as it also have a reference image, but this method initially matches alike features to predict the specific color and eventually spread it to the whole image. In iterative probabilistic relaxation method, user play a vital decision making role, as it is solely responsible to select the color code for selected parts of a gray image, hinge on a gray image gets colored. All these methods bears a similarity, and the similarity is that a prior information is provided which is usually delivered by user. A little ahead to this theory, an optimization algorithm was proposed [8] which states that the pixels of similar intensity present close to each other must have same color. Although, this method also matches the intensity of colors with the manually added color scribbles.

More often, gray scale images are one dimensional which are required to be in three dimensions, as a color image follow RGB color scheme. For a Convolutional Neural Network, the information extracted from top layers are generally mapped into the intermediate layers. In [6], output of the last layer of a CNN is not used for making the predictions, however a concept known as Hypercolumn is implemented to map information at each layer of CNN and build a fully automated image colorization system. The hypercolumn is illustrated in Figure 5

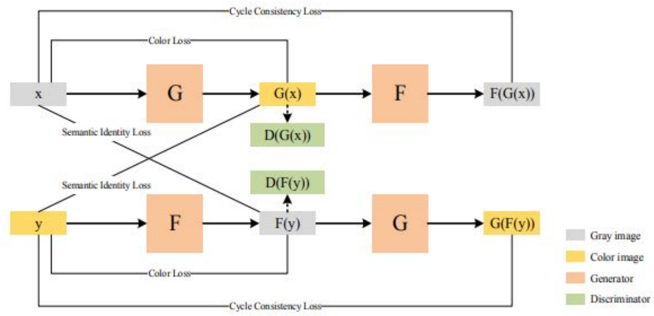
Fig. 5. Hypercolumn



As from the above figure, it is observable that, the input to the CNN is an image and is present at the bottom, while above the input image, there are feature maps of various layers of CNN. In a pixel, a hypercolumn is considered as vector from activation of all units that lie above that pixel.

Training a tradition GAN network requires paired data, which means data points with labels, however CycleGAN [20], a type of GAN, forms a cycle graph structure and does not require paired data. Inspired by this feature, the study has proposed and modified a CycleGAN-based colorization framework to suit the user needs [18]. The proposed framework is known as colorCycleGAN. Figure 6 depicts the architecture.

Fig. 6. Architecture of Color Cycle GAN

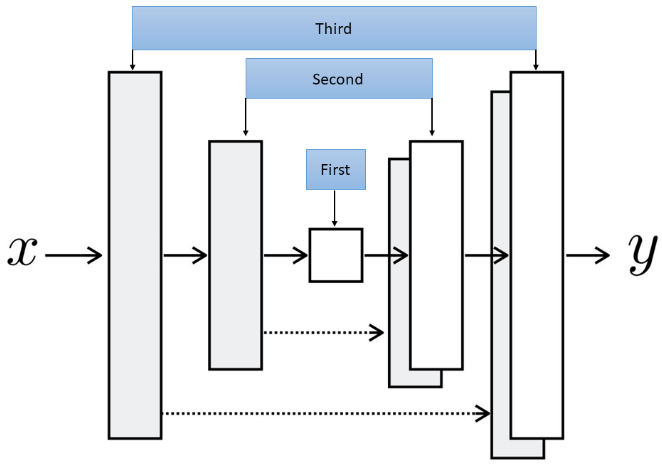


The Figure 6 shows the illustrative view of how a color cycle GAN operates. The generator network G is specifically used to map input grayscale images to colour images. A multi-layer convolutional network, similar to the U-net [15], is used. It starts with a flat convolution layer and then moves on to a lightweight 6-layer encoder-decoder framework. The U-net structure subnetwork enables low-level information to be routed across the network more quickly. A deconvolution layer then scales the learned feature map to the same spatial size as the input image. Following a convolution block, the final rgb colour image is predicted. Because the discriminant task for the discriminator network D is relatively simple, we use a patch-level discriminator to determine whether a sample is from real or generated data. To encode essential local features for classification, two convolutional blocks are used. The classification response is then obtained using a convolutional block that includes a feature transformation layer and a convolutional pixel-wise regression layer. For generator and discriminator convolution weights, spectral normalizations [10] are used (except for their respective last convolution layer).

III. Methodology

In order to achieve the main objective of this project that is to convert a gray scale image into RGB scaled image and with the help of discriminator, model would be making predictions, about an image being either real or fake, ”Tensorflow” is used to create tensors of the data, as it helps to create batches of images that makes the network run faster. The original colored images would act as the labels for the model. The generator will improves itself by generating real like images such that later discriminator produce probabilities close to 1 (True) even for generated images. Since, in order to colour the gray scale images, there will be multiple objects present in a single image, so the architecture used to implement the project is U-net architecture, due to which the generator will have structure of encoder-decoder. The encoder part of it will produce latent representations of the gray scale image, while the decoder will produce RGB image by enlarging the latent representation. A typical U-net architecture is shown in Figure 7.

Fig. 7. U-Net architecture

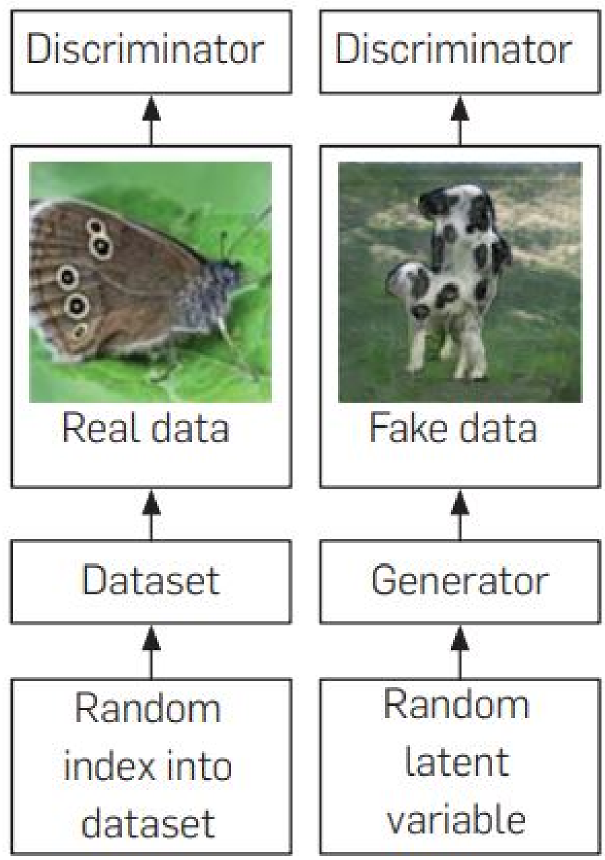


Training of Generative Adversarial Network is a two simultaneous process involving training of both generator and discriminator network. All over the process of training, generator would be continuously producing real like images which goes into discriminator along with the input colored images. Discriminator is trained in a very similar fashion as we usually train a deep learning classifier. The only difference with those classifying algorithms and discriminator is that, in this case for the process of training the discriminator, the data (images) for ”fake” class comes from a set of images that keeps on changing while training, as the generator itself learns while training and produce different images rather than fixed images as in the case of other binary classifier problem. The learning of generator while training is distinctive as there is no benchmark for its output, rather it has the responsibility to keep producing images that can fool the discriminator. For an image coming to discriminator from the set of input images, the discriminator is trained to predict it as ”real” class and the images coming from the generator, it is trained to predict it as ”fake” class. The Figure 8 shows how the mechanism works. With the help of back-propagation algorithm, it has become possible to use the derivatives of discriminator’s output with regards to discriminator’s input in order to train the generator. The role of generator is to fool the discriminator, as it is trained in such a manner that discriminator assign the input class (or images) of generator as ”real” class.

Color Cycle GAN (shown in Figure 5) is quite different from the CD-GAN shown in Figure 8, which I will be implementing in this project. A GAN consists of two neural networks, namely the generator and the discriminator, while on the other hand, cylce GAN, consists of 2 GAN’s, that is, it contains four neural networks, 2 network for generator and 2 network for discriminator. It is convenient when we have two different sets of images in our data set. Suppose, our data set contains images of dogs and cats, then under cycle GAN, one generator will be responsible for transforming dogs into cats and second generator will be responsible for transforming cats into dogs. While training this GAN, the role of discriminator would remain same as it was in the case of CD-GAN, [1] i.e., to check images coming to discriminator is real or fake. In cycle GAN, a generator has an advantage of getting the feedback not only from the discriminator but also from the second generator. The feedback from the second generator ensures the first generator that an image produced by the generator is cycle consistent, meaning that applying consecutively both generators on an image should yield a similar image.

For this project, the results obtained, contains a comparative view of original colored images that were given to the generator as an input, the gray scale image produces by the generator and the RGB image produced by the generator.

Fig. 8. Training Mechanism



IV. About the Data

The data set used for the project is a subset of the ”Standford Dogs Dataset”. The original data set contains a total of 20,580 images and these images belongs to 120 different categories of dogs breeds. ”Standford Dogs Dataset” is built using the images from ImageNet and the main purpose of this dataset is to perform fine-grained image categorization [5]. Eventually for this project, I have used 30 different categories of dog breeds, and a total of 5,129 images. Different categories of dogs used for the project are Chihuahua, Japanese_spaniel, Maltese_dog, Pekinese, Shih-Tzu, Blenheim_spaniel, Papillon, Toy_terrier, Rhodesian_ridgeback, Afghan_hound, Basset, Beagle, Bloodhound, Bluetick, Black-and-tan_coonhound, Walker_hound, English_foxhound, Redbone, Borzoi, Irish_wolfhound, Italian_greyhound, Whippet, Ibizan_hound, Norwegian_elkhound, Otterhound, Saluki, Scottish_deerhound, Weimaraner, Staffordshire_bullterrier and American_Staffordshire_terrier. Figure 9 shows one of the image from the dataset, and it belongs to ”Chihuahua” dog breed.

All these images are RGB scaled [11], which means all are colored images and have 3 dimensions. These images do not require any data pre-processing because of implementing GAN for this project, as GAN does not requires processing of data. The images are converted into tensors using tensorflow library of deep learning [13]. Batch size used for the project was 64. This is an ideal number for the batch size, earlier tried with batch size equals to 32 and 128. The number of iterations per epoch while training the model was 36 and number of epochs to which the model got trained was 15. Since, both the generator and the discriminator network trains simultaneously, thus we get a combined value of losses in each epoch. The loss value achieved after 15 epoch was 0.1030. The tensors were first given to the generator as an input, followed by discriminator. As discussed earlier, the discriminator gets images from two different sets of images, one the original set of colored images, and another from the set of colored images generated by the generator.

Fig. 9. Chihuahua Dog Breed

V. RESULTS

As discussed in the previous sections, the expected results consists of a comparative view of real and fake colored images along with the gray scale images. And another expected results would be in the form the accuracy of the model. This accuracy tells how well the model perform. Here two models executes simultaneously, one is the generator model and second is the discriminator model.

Apart from the accuracy as the evaluation metrics, there are two other metrics which could be used to evaluate the performance of a GAN model. They are Fréchet Inception Distance (FID) [12] [7] and Kernel Inception Distance (KID) [19] [14] . We have not used these two measures as the evaluation metrics because the key purpose of the project is to colorize the black and white images. And further classifying the real and fake images, which is a classification problem. Since, the above mentioned two measures are meant for different purposes like fidelity, which means evaluating the quality of images, and diversity, which means evaluating variety of images, we are not using them as the evaluation metrics. Before moving onto the results obtained in the form of accuracies, let’s discuss briefly about Fréchet Inception Distance (FID) and Kernel Inception Distance (KID).

Fréchet Inception Distance (FID):- The most common feature extractor used in classification of classes is Inception-v3 classifier. This is pre-trained on ImageNet data, a subset of which we are using as our data for the project. Under FID, the output layer of the network is disabled or cut off, and the user gets the embeddings of both real and fake images in order to extract feature distance. These two embeddings are normally distributed and could be compared using Multivariate Normal Fréchet Distance. The drawback of FID is that, since it utilises a pre-trained Inception model, it does not capture all the features. FID requires large sample size to operate because it does not have unbiased estimator leading it to give higher expected value on the smaller datasets, and lastly it is very slow to execute.

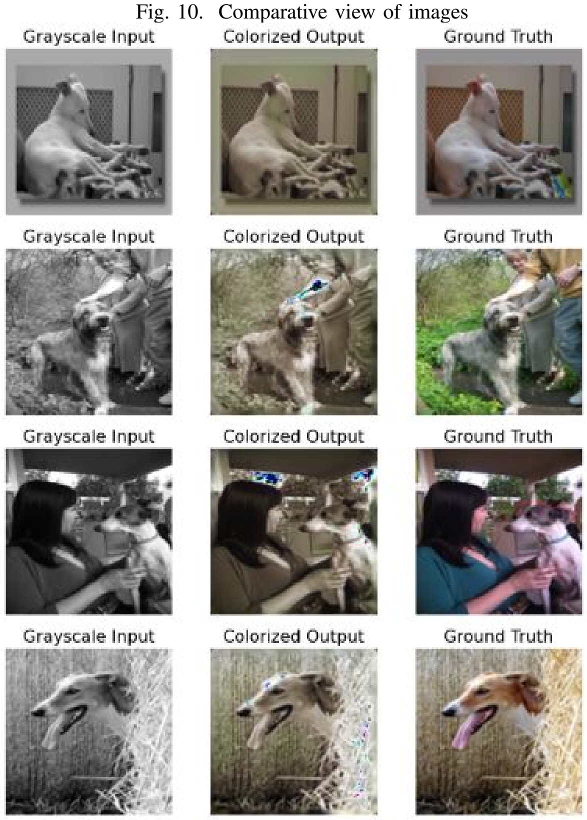
Kernel Inception Distance (KID):- KID is basically an alternative of FID. KID is compatible on smaller datasets because the expected values of KID does not depend upon the number of samples. Another advantage of KID over FID is that it is lighter to compute, it is numerically stable and quite easy to implement.

Now, let’s first discuss the performance of both model. Total number of trainable parameters for generator model is 1,406,403 and total number of trainable parameters for discriminator model is 1,907,905. Before executing the models, both models were compiled with three different parameters, optimizer, loss and evaluation metrics. The optimizer used for both the models was Adam optimizer [2]. Cross entropy loss was used to calculate generator and discriminator losses [9]. And the evaluation metrics taken into consideration for the models was Accuracy.

After executing the GAN model on the provided data set, the validation accuracy of the generator to generate real like images and validation accuracy of the discriminator for predicting whether the image is real or fake (generated by generator) is the validation accuracy of the model because the training of both architectures occurs at the same time. The validation accuracy obtained is 56%. This is the accuracy of the discriminator, which means that our model is 56% accurate in predicting the whether an image is real or fake. Despite being the fact that the accuracy value is low, the model was able to colorize the image up to a great extent in comparison to the colorization results obtained using Convolutional Neural Network in another study as shown in figure 2.

The primary expected results that fulfils the scope of the project is the comparative view. Figure 10 represents the output.

Fig. 10. Comparative view of images

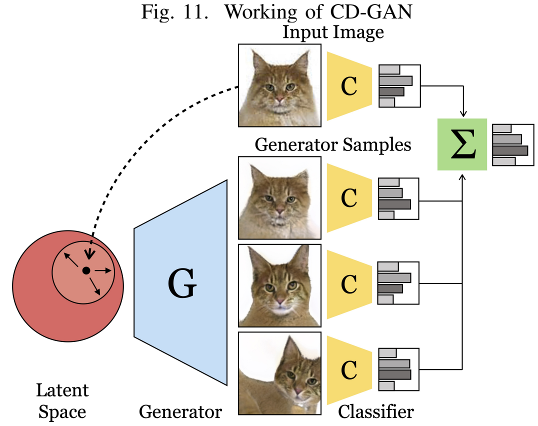


The first column shows the gray scale images that were produced by the generator. These images were obtained after converting the colored input images into black and white. The second column shows the colored images that were produced by the generator. These images were obtained after the previous step. The third column shows the actual colored images that were given to the generator as the input. Taking a deep look at the output, it is a remarkable thing that the colored images generated from the generator are colorized effectively. It means that the generator worked properly, and fulfils its role of converting colored images into black and white and then colorizing it. Also, from the comparative view of the output makes it easier for a layman to understand how the complete process of GAN works.

VI. Discussion / Conclusions

Generative Adversarial Networks is a very special kind of deep learning models, and out of the other models it is basically based upon game theory. It has drastically overshadowed other deep learning algorithms in terms of generating artificial data especially images. These artificially generated images have a realistic nature which makes it, quite difficult for a user to differentiate between an actual image and the image generated by the GAN's. For this project, we have to do image colorization, into which GAN's did a fantastic job as it successfully converted the colored images into gray scale and further automatically generates the colored pattern of the gray image. The whole scenario of this is illustrated in Figure 11.

Fig. 11. Working of CD-GAN



In figure 11, the latent space should be considered as the dataset of our project, which contains all the colored images and is used to send it to the generator as an input. "G" is the generator. The input image is given to the generator, the three images shown in front of the generator are the three different colored images obtained in three different epochs. Thereafter, GAN is implemented as the classifier. The sigma symbol shows the discriminator, to which both the original colored input image and the colorized outputs of generator are given as the input, which makes a prediction at the last. This prediction differentiates the real colored image and fake (generated) colored image.

The backbone of the whole project was selection of dataset for the GAN model to train. The training dataset was required to be a colorized image dataset, which should have contain some complex patterns in order to build a robust model. The dataset used for this project is a set of colored images of 30 different dog breeds, which makes it complex in nature. The behaviour of the dataset is determined on the basis of how clean the data is. The dataset used is a standardised dataset taken from Standford Dogs Dataset, which is a subset of ImageNet dataset, that's why the images present into it is clean, in proper shape and size, in proper alignment which eventually makes it a perfect dataset to use [3].

In Generative Adversarial Networks, it is not necessary to do data pre-processing, because GAN's are capable of handling the flaws of data, and provides outstanding results on its own. It basically extract important features from its architecture inclusive of generator build with the help of

CNN and discriminator. GAN's are appropriate for facial image generation and recognition. In the case when we have to execute a vital part of image processing [16] i.e., facial emotion or expression recognition, GAN's are powerful tool to multiply the dataset, in the case of deploying a model with less data. It can generate enormously efficient facial images, which are very similar to the actual facial images present in the dataset. Similarly, in the scope of this project, the GAN does generates colored images. It does not multiplies the number of images present in the dataset, because it wasn't required.

The main conclusion drawn from this project is the fact that it is feasible and practicable to convert a gray scale image into colored image, with the help of deep learning models. It is a great contribution to the world as it helps layman to live old black and white images as colored images. Also, this project shows how the artificial intelligence has shown a significant rise into the lives of human beings to make their life easier.

The apex motivation of deploying the project is fulfilled, however there is a scope of improvement in the process of colorization. Increasing the size of dataset may improve the accuracy of the model. In comparison to the Convolutional Neural Network, GAN is less accurate in predicting the originality of the image, because in CNN, the feature is extracted from every layer of the CNN, but in GAN the prediction is made only with the help of the last layer. CNN is often used in classification problems but not GAN's. However, this project is basically purposed to solve the problem of colorization of images, which makes GAN better than CNN becaus in the case of generating the images, CNN do not work as per the expectations.

VII. Future Work

The scope of this project could be extended with the inclusion of videos. This project has shown the image colorization using GAN's on the given images, however in a real world we quite often uses videos and abundance of video data make it the future scope of this project. There are a great quantity of black and white videos performed in the past, which we can make it colored with the help of this technique.

Secondly, there is a scope of further research and development in the area of Generative Adversarial Networks, comparing different types of GAN's on the basis of how they perform on colorizing images and videos. The parameter to be taken into consideration should be accuracy only, as this project is not solely a binary classification problem, rather it is a project into which we are getting solid outputs which should match out desires. However, in the case where we do not have to get the results as the comparative view of images but rather focusing on generating images and would be analysing fidelity and diversity of images, then we can use Fréchet Inception Distance (FID) and Kernel Inception Distance (KID) as the evaluation metrics.

VIII. Acknowledgement

I would extend my extreme gratitude towards my supervisor Sebastian Berns for his regular monitoring, advice on dataset selection, representation of output obtained, practical usage of GAN's, and much more different aspects while performing the model and writing this paper.

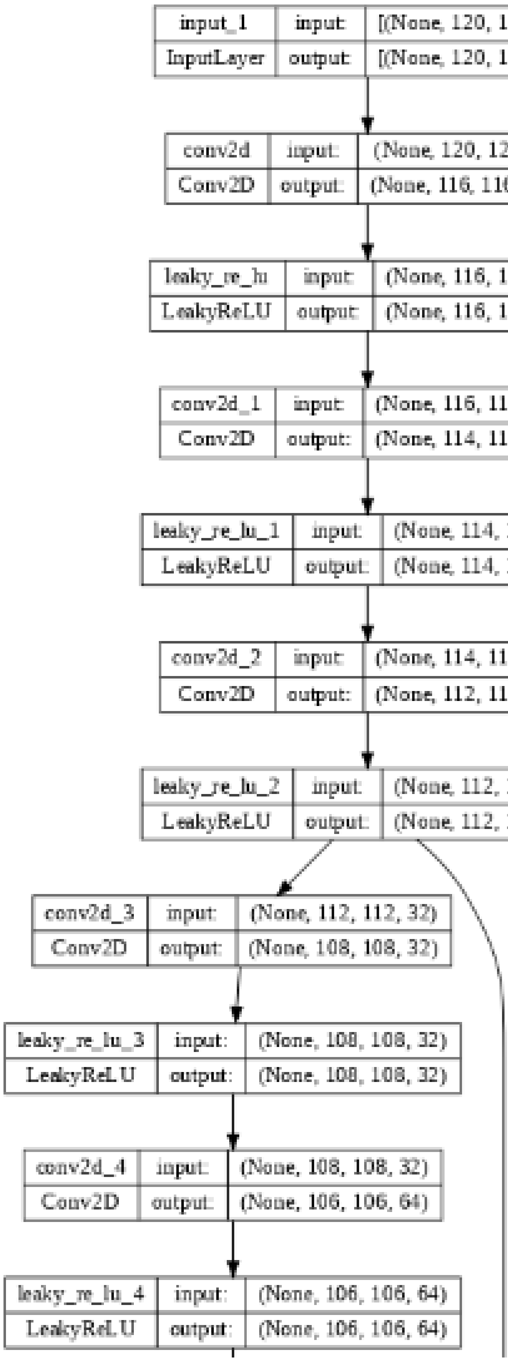
References

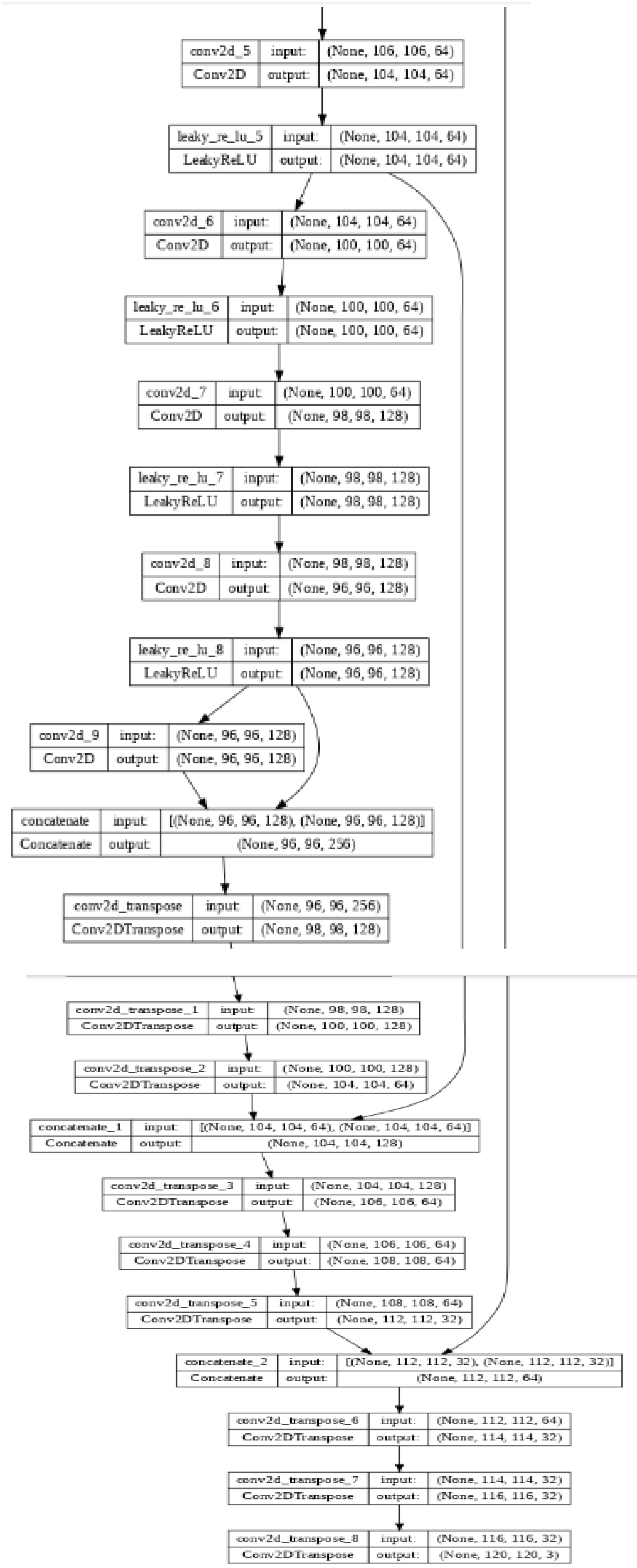
[1] Kancharagunta Kishan Babu and Shiv Ram Dubey. Cdgan: Cyclic discriminative generative adversarial networks for image-to-image transformation. *Journal of Visual Communication and Image Representation*, 82:103382, 2022.
[2] Kiwook Bae, Heechang Ryu, and Hayong Shin. Does adam optimizer keep close to the optimal point? *arXiv preprint arXiv:1911.00289*, 2019.
[3] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.
[4] Michal Kawulok and Bogdan Smolka. Competitive image colorization. In *2010 IEEE International Conference on Image Processing*, pages 405–408. IEEE, 2010.
[5] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Li Fei-Fei. Novel dataset for fine-grained image categorization. In *First Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition*, Colorado Springs, CO, June 2011.
[6] Leila Kiani, Masoud Saeed, and Hossein Nezamabadi-pour. Image colorization using generative adversarial networks and transfer learning. In *2020 International Conference on Machine Vision and Image Processing (MVIP)*, pages 1–6. IEEE, 2020.
[7] Tuomas Kynkäänniemi, Tero Karras, Miika Aittala, Timo Aila, and Jaakko Lehtinen. The role of imagenet classes in fr\'echet inception distance. *arXiv preprint arXiv:2203.06026*, 2022.
[8] Anat Levin, Dani Lischinski, and Yair Weiss. Colorization using optimization. In *ACM SIGGRAPH 2004 Papers*, pages 689–694. 2004.
[9] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, and Zhen Wang. Multi-class generative adversarial networks with the l2 loss function. *arXiv preprint arXiv:1611.04076*, 5:1057–7149, 2016.
[10] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.
[11] Soumya Ranjan Nayak, Jibitesh Mishra, Asimananda Khandual, and Gopinath Palai. Fractal dimension of rgb color images. *Optik*, 162:196–205, 2018.
[12] Artem Obukhov and Mikhail Krasnyanskiy. Quality assessment method for gan based on modified metrics inception score and fréchet inception distance. In *Proceedings of the Computational Methods in Systems and Software*, pages 102–114. Springer, 2020.
[13] Bo Pang, Erik Nijkamp, and Ying Nian Wu. Deep learning with tensorflow: A review. *Journal of Educational and Behavioral Statistics*, 45(2):227–248, 2020.
[14] Suman Ravuri and Oriol Vinyals. Classification accuracy score for conditional generative models. *Advances in neural information processing systems*, 32, 2019.
[15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.
[16] Pilar Rosado, Rubén Fernández, and Ferran Reverter. Gans and artificial facial expressions in synthetic portraits. *Big Data and Cognitive Computing*, 5(4):63, 2021.
[17] Yucheng Tang, Ho Hin Lee, Yuchen Xu, Olivia Tang, Yunqiang Chen, Dashan Gao, Shizhong Han, Riqiang Gao, Camilo Bermudez, Michael R Savona, et al. Contrast phase classification with a generative adversarial network. In *Medical Imaging 2020: Image Processing*, volume 11313, pages 220–227. SPIE, 2020.
[18] Yuxuan Xiao, Aiwen Jiang, Changhong Liu, and Mingwen Wang. Single image colorization via modified cyclegan. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 3247–3251. IEEE, 2019.
[19] Matthew Yates, Glen Hart, Robert Houghton, Mercedes Torres Torres, and Michael Pound. Evaluation of synthetic aerial imagery using unconditional generative adversarial networks. *ISPRS Journal of Photogrammetry and Remote Sensing*, 190:231–251, 2022.
[20] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2242–2251, 2017.

APPENDIX

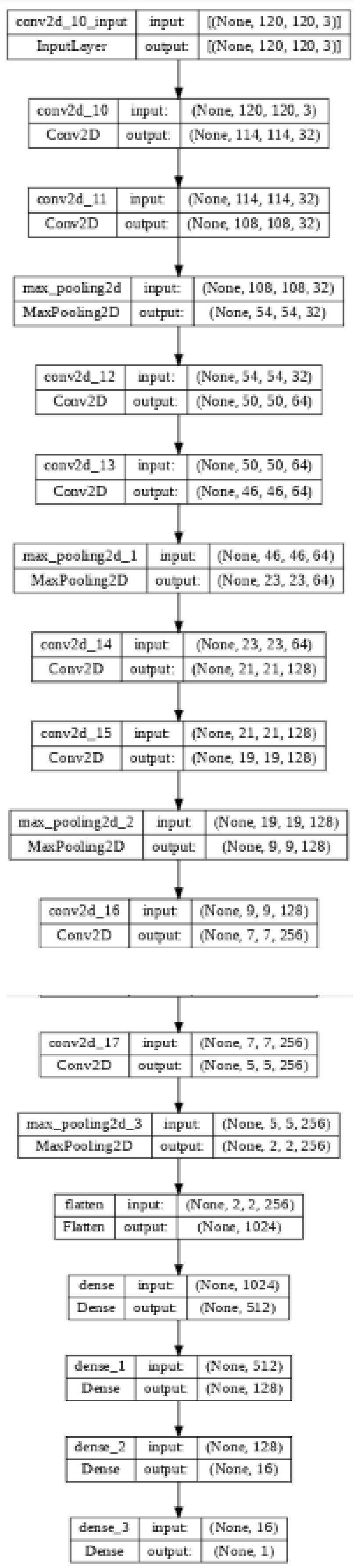
This part of the project is being used to show the structure of the generator and discriminator networks of our Generative Adversarial Network model.

Firstly, the Generator Network:




conv2d_5 | input: | (None, 106, 106, 64)
Conv2D | output: | (None, 104, 104, 64)
leaky_re_lu_5 | input: | (None, 104, 104, 64)
LeakyReLU | output: | (None, 104, 104, 64)
conv2d_6 | input: | (None, 104, 104, 64)
Conv2D | output: | (None, 100, 100, 64)
leaky_re_lu_6 | input: | (None, 100, 100, 64)
LeakyReLU | output: | (None, 100, 100, 64)
conv2d_7 | input: | (None, 100, 100, 64)
Conv2D | output: | (None, 98, 98, 128)
leaky_re_lu_7 | input: | (None, 98, 98, 128)
LeakyReLU | output: | (None, 98, 98, 128)
conv2d_8 | input: | (None, 98, 98, 128)
Conv2D | output: | (None, 96, 96, 128)
leaky_re_lu_8 | input: | (None, 96, 96, 128)
LeakyReLU | output: | (None, 96, 96, 128)
conv2d_9 | input: | (None, 96, 96, 128)
Conv2D | output: | (None, 96, 96, 128)
concatenate | input: | [(None, 96, 96, 128), (None, 96, 96, 128)]
Concatenate | output: | (None, 96, 96, 256)
conv2d_transpose | input: | (None, 96, 96, 256)
Conv2DTranspose | output: | (None, 98, 98, 128)
conv2d_transpose_1 | input: | (None, 98, 98, 128)
Conv2DTranspose | output: | (None, 100, 100, 128)
conv2d_transpose_2 | input: | (None, 100, 100, 128)
Conv2DTranspose | output: | (None, 104, 104, 64)
concatenate_1 | input: | [(None, 104, 104, 64), (None, 104, 104, 64)]
Concatenate | output: | (None, 104, 104, 128)
conv2d_transpose_3 | input: | (None, 104, 104, 128)
Conv2DTranspose | output: | (None, 106, 106, 64)
conv2d_transpose_4 | input: | (None, 106, 106, 64)
Conv2DTranspose | output: | (None, 108, 108, 64)
conv2d_transpose_5 | input: | (None, 108, 108, 64)
Conv2DTranspose | output: | (None, 112, 112, 32)
concatenate_2 | input: | [(None, 112, 112, 32), (None, 112, 112, 32)]
Concatenate | output: | (None, 112, 112, 64)
conv2d_transpose_6 | input: | (None, 112, 112, 64)
Conv2DTranspose | output: | (None, 114, 114, 32)
conv2d_transpose_7 | input: | (None, 114, 114, 32)
Conv2DTranspose | output: | (None, 116, 116, 32)
conv2d_transpose_8 | input: | (None, 116, 116, 32)
Conv2DTranspose | output: | (None, 120, 120, 3)

Secondly, the Discriminator Network:



M.Sc. Project – Reflective Essay

Project Title:	Image Colorization using Generative Adversarial Networks
Student Name:	Ritwik Sinha
Student ID:	220016977
Supervisor name:	Sebastian Berns
Programme of Study:	M.Sc. Big Data Science with Machine Learning Systems

The study coveted to address the challenge of understanding the unstructured data in the form of images by progressively performing research and development methods in order to colorize black and white images with the help of a deep learning algorithm known as Generative Adversarial Networks (GAN's), and also classifying real coloured image and fake (or computer generated or AI generated) coloured images. The study briefly introduces with various other methods of colorizing black and white images into coloured images. Along with that, it has also described why GAN's are the best network to implement on this study.

Approach:-

The approach was a bit complicated as for doing image colorization, the main purpose is not to give black and white image as an input to the first network of the model i.e., the generator, the main purpose is to colorize a black and white image which itself would be created from a set of coloured images given as an input to the generator. The generator is designed in such a manner that it will take coloured image, convert it into black and white and then colorize it. This complication is introduced in this study because I wanted to perform a study where the next step after the colorization should be classifying original coloured images and coloured images generated by the generator, so that, the problem of identifying fake images that are present in enormous number on the internet could be solved. These fake images tends to misguide a user from the originality of the product. This usually happens in the case of E-commerce websites, Videography websites, a website loaded with humungous images etc. As the discriminator classifies original and fake images, the study will eventually help the user to understand how an AI generated image is different from the original image.

There are two types of output that I achieved in this study. The first output depicts the accuracy of my GAN model, this accuracy shows how accurately my model has classified real and fake images. This is a GAN model, we should aim for best in class generated images, colorizing it, extracting features etc. This eventually gets satisfies by the second output obtained from the study. This output shows a comparative view of original coloured images, black and white images generated from the generator and the colourised images of black and white images that even by generator.

The last but not the least challenge that I faced while implementing this approach was to understand the behaviour of generator. As the quality of generator is that, it learns from itself while training, and improves the quality of generated images during the time of training. This is so because the training of generator and discriminator takes place simultaneously in GAN. Generator tends to fool the discriminator and vice versa. At one side I was abide to improve the performance of generator and at the other go I was abide to improve the performance of discriminator. In order to fulfil the scope of the project it was necessary that both the networks perform well. This happened to a great extent and has given good results that eventually satisfies the purpose of study.

Practical challenges:-

As a student of Data Science, in addition to writing a research paper I have to demonstrate practical view of the problem. Which means, implementing the problem statement with the help of a high level programming language. I used Python as the programming language in order to implement the problem. There were lots of challenges through which I went to while doing the implementation. Let's discuss some of the major challenges which I faced:

1. **Challenge to select the platform**: - The first challenge was to choose a perfect python environment in which I could do the coding. I have chosen Google Colaboratory as the platform. The best part of choosing Google Colaboratory is that, it can be linked to the Google Drive. Via Google Drive, I was able to easily call the dataset, which I stored in my Google Drive, on my Colaboratory Notebook. Addition to it, the Colaboratory provides an extra advantage of built in python libraries. I was not required to install python libraries to execute the code as Colaboratory has those libraries already installed into it.

   Colaboratory also has provided an advantage of utilising free GPU service from Google. As my local machine does not have GPU in it, and in order to implement this project, I was abide to use GPU so that the whole model could be executed in less amount of time. I easily utilised free GPU of Google Colaboratory. The only disadvantage with this free service, is that, we can use it for a limited amount in a day, after which we can't use it. And if we want to use GPU more, we have to purchase the subscription of Google Colaboratory Pro or Google Colaboratory Pro+.

2. **Challenge to select the type of Deep Learning Model**: - Colorization of black and white images is a genuine task of deep learning. Choosing the right deep learning model was a major task. Lots of researches that has occurred in the past regarding the Image colorization has used Convolutional Neural Network (CNN). However, I wanted to implement Generative Adversarial Networks (GAN's). It was not possible to completely use GAN for the project as one of the scope of the project was to classifying the real and fake coloured images, that's why this project has utilised CNN also for the issue of classification in the form of discriminator.

   Once, I decided to use GAN as the deep learning model, the next challenge was to execute the model, as it was my first time that I was building a model in which two neural networks should run simultaneously (here generator and discriminator). It was so because both the networks were abide to fool each other and would try to diminish the accuracy of each other and improving their own while training. This was the most challenging part, and I believe that the results obtained were well as per the expectations.

3. **Challenge of choosing the dataset**: - From the complete cycle of deploying a machine or deep learning model, choosing of appropriate dataset takes a lot time, and I faced this challenge as well. For this project, I wished to have a dataset which contains coloured images and multiple classes. At the beginning of the project, I implemented the whole model on a dataset which I downloaded randomly from internet. It wasn't a common dataset like FMNIST, KMNIST, CIFAR 10, ImageNet etc., it was a very basic dataset having images of natural beauty, human beings and vehicles. After I implemented the whole model, I realised I wasn't getting a satisfactory accuracy score, and I thought to change the dataset. It was a great challenge to implement the project on new dataset, because the network which build for previous dataset, started performing much poorer on the new datasets. After trying to implement the model on different datasets, I found a dataset named as Stanford Dogs Dataset, which contains 120 different breeds of dogs. This dataset is basically a subset of very famous ImageNet dataset. This dataset is originally developed in order to perform a task of fine-grained image categorization. However, I utilised this dataset for my project of image colorization. This particular dataset gives me desired results. It took around 40% time of whole project for selecting the dataset.

4. **Challenge of searching research papers for Literature Review**: - While writing my own project paper, it was a challenge to look for journals or papers that were published in the past for the task of image colorization of black and white images. Since, it was purely implementation based project, so there were immense number of study performed in the past, with the help of different machine or deep learning models, as well as, with the help of manual methods of image colorization.

5. **Challenge of inserting images in Latex**: - I was provided with a Latex Template from my university for writing the paper. When I started writing the paper, it was quite obvious to insert images in the paper so as to make the content relatively simple for the readers. As image is the best form of visualising something. Overleaf is a platform where we can use the Latex template, I faced the challenge of inserting the images into it. As whenever I tried to insert an image in between the content, the whole alignment of the paper gets disturbed. Sometimes, the size of the image becomes an issue, sometimes the quality of image becomes an issue (in the case when I stretch the image, the image becomes blurry).

Personal Development:-

It was a great opportunity for me to perform and complete this study irrespective of several challenges that comes across the way of finishing it. The aim and purpose of the study gets satisfactorily fulfilled and I obtained the desired results. I was a data science enthusiast since my under graduation where I chose Data Science as specialisation, I got to know the basics of data science there and in order to excel more I joined the MSc. Big Data Science with Machine Learning Systems in January 2022. By joining this course, I attained a very deep understanding of the core concepts of data science, such as Machine Learning, Deep Learning, Big Data, etc. At the end of the course, i.e., third semester, I have to do a research project. I chose Sebastian Berns as my supervisor and he guided me throughout the project. I must acknowledge him for helping me in choosing the right dataset and implementing GAN model on it. As GAN is a new deep learning model at the beginning of the project, he guided me a lot to learn the concepts of GAN's, and at the end of this project I must say that I have attained a sound knowledge of GAN's. Not only performing image colorization of black and white images using GAN's, I also learnt a lot about GAN's, how it is being utilised for generating artificial images and it genuinely creates real like images. Since, any deep learning algorithm requires a good volume of data in order to perform well, but in the real world it is difficult to get a dataset with huge volume of data. One option is to use pre-trained models, but the pre-trained models has a disadvantage of not performing well as the dataset used in the pre-trained models, may not match the features and characteristics of dataset used in original project. Hence, GAN's helps to generate data, and solves the issue of lack of data, and successfully overcomes the disadvantage of pre-trained models. At whole, this project has improved my quality of writing, research and development skills.

REFERENCES:-

[1] On the evaluation of Generative Adversarial Networks.

https://towardsdatascience.com/on-the-evaluation-of-generative-adversarial-networks-b056ddcdfd3a#:~:text=The%20most%20common%20metrics%20to,diversity%20(variety%20of%20images).&text=In%20order%20to%20compare%20real,be%20used%20as%20feature%20extractors.

[2] Notes on Convolutional Neural Networks.

https://web-archive.southampton.ac.uk/cogprints.org/5869/1/cnn_tutorial.pdf

[3] Top 6 Metrics To Monitor The Performance Of GANs.

https://analyticsindiamag.com/top-6-metrics-to-monitor-the-performance-of-gans/

[4] Stanford Dogs Dataset.

http://vision.stanford.edu/aditya86/ImageNetDogs/

[5] #07 Hyperparameter Tuning: how to improve model accuracy drastically.

https://medium.com/shortcutnlp/07-hyperparameter-tuning-a-final-method-to-improve-model-accuracy-b98ba860f2a6

[6] 6 GAN Architectures You Really Should Know.

https://neptune.ai/blog/6-gan-architectures

[7] Dataset.from_tensors versus Dataset.from_tensor_slices.

https://stackoverflow.com/questions/49579684/what-is-the-difference-between-dataset-from-tensors-and-dataset-from-tensor-slic

[8] Advanced Courses: Machine Learning and Generative Models.

https://developers.google.com/machine-learning/gan