International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 6, Issue 11, ISSN No. 2455-2143, Pages 203-212
Published Online March 2022 in IJEAST (http://www.ijeast.com)

COLORIZATION OF GRAYSCALE IMAGES IN DEEP LEARNING

Angelica D Pyngrope[1], Dr. Pawan Kumar[2]
[1]MCA Student, [2]ASST. Professor, School of CS and IT, Dept of MCA,
JAIN (Deemed-to-be University), Jayanagar 9th Block, Bangalore, Karnataka (IN)

***Abstract*—One of the most exciting Deep Learning applications is color grading black and white pictures. Coloring a grayscale image is a simple exercise for the human mind in general; we learn to fill in missing shades in colouring books from a young age. This task used to necessitate a large number of people involvement and complex tasks, however in the recent years, thanks to AI and Deep Learning, the entire process can be automated from start to finish. In this paper, I looked at a number of publications that presented various AI and Deep Learning methodologies. I'll go over each approach and tactic that has been utilised to investigate different Neural Networks in Deep Learning and how they may be used to achieve incredible outcomes. Throughout this paper, I'll start by incorporating whatever the researchers achieved, and I'll add a wholly separate generator model as well as some changes to the training technique, which will significantly reduce the overall size of the required dataset whilst producing remarkable outcomes. A GAN based model is proposed and trained on the COCO dataset. A new experimental model has been introduced which marks as the final model of our system. We notice a slight decrease in the loss function in the final model in comparison to the first model. The baseline model computed a loss of 7.6 % and the final model computed a loss of 6.7% which denotes that the final model produces results with higher accuracy.**

***Keywords*— Colorizing black and white images, Deep Learning, AI, GAN.**

I. INTRODUCTION

The act of taking an input grayscale (black and white) image and turning it into an output colourized image that represents the input's conceptual colours and tones. For example, an object in the image with the colour yellow must be identified; otherwise, the model will colour it blue.

Deep learningis a machine learning technique that uses artificial neural networks and representation learning. Learning can occur in supervised, semi-supervised, or unsupervised conditions.

When photography was first established, only black and white photos were available due to technological restrictions. Colour photography, on the other hand, is now commonplace. There are numerous recollections and links between the present and the past when it comes to historical photography. Converting them to coloured versions would be more fascinating in terms of enhancing hidden meanings and making them more visually appealing. Manual or Photoshop colorization was used, which took a long time. In recent years, many Deep Learning-based colorization techniques have been proposed. When a colour image is transformed to a grayscale version, information is lost across dimensions, which causes the colorization problem. Some solutions took a classification approach to the problem, while others took a regression method.

In order to evaluate these approaches, I present a system of what the authors have done, then incorporate a whole generator model and tweak their training strategy.

Convolutional neural networks (CNNs) have become the universal powerhouse driving a broad large number of images prediction tasks, and the community already has some portion of profit progress in this regard. While CNNs learn to minimize a loss function - an objective that evaluates the performance in terms of quality a lot of human effort was put into developing efficient losses, regardless of the fact that the process of learning is replaced by automation. To put it another way, we still need to tell CNN what we want it to diminish.It would have been ideal if we can just alternatively give merely a slightly elevated aim, such as making the result undetectable from actuality, but then have the system automatically develop a loss function that fulfils thisneed. Remarkably, our recently proposed Generative Adversarial Networks (GANs) achieve precisely that. GANs generate a loss function which attempts to distinguish whether the produced picture is actual or false even while learning a generative model to reduce the loss. Photos that are fuzzy or appear to somehow be false would not be accepted. GANs may be used for a variety of tasks which would normally demand quite diverse types of loss functions since they train a loss which adjusts to the data.We investigate GANs in the dependent environment in this study. Conditional GANs also known as CGans train a conditional generative model in the same way as GANs generate a generative predictive representation of data.Within the recent years, GANs have garnered considerable attention, many of the strategies we look at with this work were already developed. However, previous

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 6, Issue 11, ISSN No. 2455-2143, Pages 203-212
Published Online March 2022 in IJEAST (http://www.ijeast.com)

work concentrated on advanced systems, which is still unknown whether successful photo conditional GANs may be as an overall colorization technique.

The purpose of this research is to investigate picture colorization as a justification job for learning visual features. The technique of adding colour to a previously black-and-white image is known as image colorization. Since each image can be decomposed into its grey and colour elements, it fits under the group of self-supervised assignments. When operating on the Lab colour space, for instance, the L-channel is employed as the picture colorization model's input, while the a and b channels are the fake identifiers that the model must learn to identify.

The rest of the study is arranged as follows. Chapter II examines some cutting-edge research in the field. Secondly, in Chapter III, we go over the suggested picture colorization approaches and explain the training procedure. In Chapter IV, we go through the datasets that were employed, the construction, the experimentation setting, and the conclusions that were obtained. In Chapter V, we examine the findings and provide recommendations for further research. Lastly, Chapter VI brings the paper to a close.

II. RELATED WORK

Numerous fields of computer vision research are interconnected and integrated into the colorization process. Colorization research papers vary considerably due to the diversity of problem-solving methods proposed. The creativity and variety of approaches make characterizing various colorization methods incredibly challenging.

The journey of deploying a colorization model started after numerous research, obtaining insights from community forums, experimentation and evaluating each methodology proposed by different authors. After a long process of analyzing a wide variety of works I finally succeeded putting together some the of techniques of their work and apply very few modifications to their training process. After consulting my project guide about how to go about with the idea, I managed to publish a literature survey describing each paper that have implemented similar systems. Through this survey, I am analyzing and comparing each work contributed by the authors. Furthermore, I was able to publish the work[1] tilted “Colorization of Black and White Images: A Survey”. The following review is a continuation of the prior work which I have submitted.

Only till recent times, the majority of the current papers classified colorization methods determined by the amount of user participation in problem solving and the method of obtaining the necessary data[2], [3], [4], [5], [6], [7], [8]. The scribbled-based techniques was and methods based on examples were introduced. The source i.e., referenced images for the example-based methods could be obtained manually or automatically. However, the manner of this type of classification is obsolete. Due to the advances of Deep Learning methods, there was need to separate the source images. Therefore, the newly proposed papers that used colorization methods as a deep learning approach utilized incomparably large training data[9], [10].Hence, the need for separating into scribble-based, example-based, and learning-based methods was noted. Anwar et al. [11]came up with the suggestion of dividing the learning methods into several categories such as the type of domain being used, auxiliary input, neural network and finally, the final outcome.

In recent years, a variety of deep learning-based colorization techniques have been developed. The Colorful Image Colorization paper took a classification approach to the problem, and they took into account the problem's uncertainty (for example, a vehicle in the image can take on many distinct and genuine colours, and we can't be certain about any of them); however, another paper took a regression approach to the problem, with a few system modifications. Each approach has advantages and disadvantages, and I will describe how each differs from the other in terms of the strategies and techniques used and the accuracy of their model.

Richard Zhang et al.,[12]approached the problem by using a feed forward pass in a CNN and has stated it as a multinomial classification. He utilized the Image Net dataset and extracted only the first few 10,000 images for training and testing. A "colorization Turing test" was being used to evaluate the algorithm, which required human volunteers. To signify predictions using a natural objective function, the objective function is optimised using the CIE Lab colour space model in the Euclidean Loss L_2 between predicted and ground truth colors.

Loss function:

$$L_{cl}(\hat{\mathbf{Z}}, \mathbf{Z}) = -\sum_{h,w} v(\mathbf{Z}_{h,w}) \sum_{q} \mathbf{Z}_{h,w,q} \log(\hat{\mathbf{Z}}_{h,w,q})$$

A mapping from a given input to a probability distribution on all feasible colours, where Q is the number of quantized ab values. A function that transforms ground truth Y to vector Z using a gentle encoding approach is defined to compare predictions against ground truth. Finally, a multinomial cross entropy Loss $L_{cl}$is defined as :

$$L_{cl}(\hat{\mathbf{Z}}, \mathbf{Z}) = -\sum_{h,w} v(\mathbf{Z}_{h,w}) \sum_{q} \mathbf{Z}_{h,w,q} \log(\hat{\mathbf{Z}}_{h,w,q})$$

where v (·) is a re-balancing concept based on color-class scarcity that can be used to re-balance the loss.

Phillip Isola, et al.,[13]Phillip applied conditional adversarial networks to image-to-image translation challenges as a general-purpose solution. Their system comprises of a few other techniques such as synthesizing photos from label maps, reconstructing objects from edge

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 6, Issue 11, ISSN No. 2455-2143, Pages 203-212
Published Online March 2022 in IJEAST (http://www.ijeast.com)

maps, and colorizing images, among other tasks. He mentions some of the drawbacks of utilising the CNN method for colorization, such as the need to minimise the Euclidean distance in[12],between predicted and ground truth colors, The issue is that the Euclidean distance can only be decreased if all of the outputs have an average value, which will further blur the image. As a result, he chose a conditional GAN that will learn a conditional generative model, distinguishing it from ordinary GAN models. Various researchers have already experimented with GANs in the past., [14]illustrates the usage of the model to learn a multi-modal model as well as generation of image tags. However, in [13],the usage of the GAN network is solely for the purpose of image translations. The system architecture includes a generator-discriminator in which both have different roles such as, a U-net architecture is used for the generator and as for the discriminator, a Patch GAN classifier is used which will penalize structure at the scale of image patches. The modules used by the generator-discriminator is of the form convolution-Batch Norm-ReLu. Another important aspect is the level of information between the input and output, resulting in a bottleneck as well. The addition of a skip connection to the U-Net architecture in the generator has been proven to tackle this. The configuration of a GAN neural network can be illustrated through the figure below which will describe how the paper utilized the algorithm:

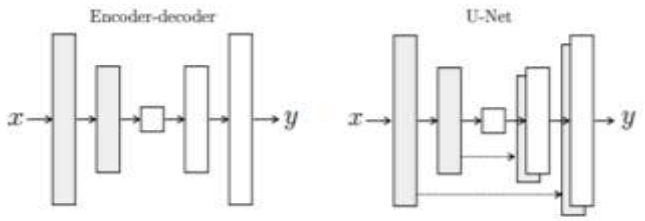

Fig 2.1 Two choices for the architecture of the generator. U-net architecture. [13]

In Fig. 2.1, the GAN learns a mapping from observed image x and random noise vector z, to y, G : {x, z} → y. During training and test time, noise is only delivered in the form of dropouts, which is distributed to numerous layers of the generator. Considering the dropout noise, the output of the nets has relatively moderate randomness.

The goal of Madhab et a.,[15]colorization was to protect Nepal's historical culture by retaining its uniqueness. In addition, the study recommended employing a CNN in conjunction with an Inception-RestnetV2 and the RGB colour model for pattern recognition using the back propagation method. To extract low-level information from the input image, the network also employs an encoder-decoder architecture. A self-generated dataset of 1,200 ancient and historical photographs of Nepal with a resolution of 256x256 pixels was employed. The MSE (Mean Squared Error) and the PSNR (Percentage of Squared Error) are the two loss functions that have been applied (Peak-Signal-to-Noise-Ratio). The model's validation is combined with a subjective value as the MOS to assess colorization accuracy (Mean Opinion Score).

Y. Morimoto, et al., [16]proposed an automatic colorization system based on information from a scene structure that used one million photos. A gist scene descriptor, which is a feature vector to characterize the global scene in lower dimension, was utilised to discover the source image. In [17], through examples, the approach learns to colourize. A LEARCH framework is utilised to train a quadratic objective function that is analogous to a Gaussian random field in the chromaticity maps. The goal function coefficients are trained on image features using a random forest.

Convolutional neural network approaches have recently been applied to the problem of colorization by a number of academics. As noted in the study, the CNN(Convolutional Neural Networks) has been shown to be the most commonly utilised algorithm

There are various aspects to their proposed methodology that must be examined.:

- The network is trained using a multinomial cross entropy loss function weighted by colour rarity.
- The approach works well on a variety of photos and captures the multimodal aspect of pixel colour; nonetheless, colour bleeding defects occur occasionally, as they do with many other colorization algorithms.
- Image semantics are well captured by convolutional neural networks. The weights for each cost element, however, are dependent on the input photographs, making the model difficult to generalise to a large number of images. Even though the computation is done on GPU, it is difficult to achieve a real-time colorization experience when pixels are generated with reasonable based optimization.

III. METHOD

3.1GANs (Generative Adversarial Network)

Generative adversarial networks, or GANs for short, are an effective deep learning approach for developing generative models. Firstly, we sample some noise z using a normal or uniform distribution. With z as an input, we use a generator **G** to create an image x (x=G(z)).

The discriminator is used in the training process to ensure that the generator learns the sample distribution. The discriminator uses both the generator's created samples and real samples from the dataset to train. The discriminator is then taught to distinguish between manufactured and actual samples as a classifier. The generator is also taught by fooling the discriminator during the discriminator's training, which results in learning the dataset's features. As a result,

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 6, Issue 11, ISSN No. 2455-2143, Pages 203-212
Published Online March 2022 in IJEAST (http://www.ijeast.com)

GAN training can be compared to a game between the discriminator and the generator.
A GAN generator model is taught using a second model called a discriminator that learns to classify images as real or produced, unlike conventional deep learning neural network models that are trained using a loss function until convergence. To maintain equilibrium, both the generator and discriminator models are taught jointly.

3.2 Objective
As previously stated, the discriminator acts as a binary classifier which means that when we provide the model actual data, the model should provide high probability for real data and low probability for fake data, which will be the generator's output.
Taking a look at the proposed system, the generator version takes a grayscale photograph (1-channel picture) and produces a 2-channel image, a channel for *a and any other for *b. The discriminator, takes those produced channels and concatenates them with the enter grayscale image and comes to a decision whether or not this new 3-channel photo is faux or actual. Of course, the discriminator also desires to look at some actual pictures i.e., three-channel images once more in Lab color area that are not produced by way of the generator and should learn that they may be real. This particular condition that is stated relates the fact that the monochrome image which each the generator and discriminator see is the condition that we offer to both models in our GAN and count on that the they take this condition into consideration.
Verifying the mathematical expression, the objective function of the GAN can be described as:

$$\mathcal{L}_{cGAN}(G,D) = \mathbb{E}_{x,y}[\log D(x,y)] + \mathbb{E}_{x,z}[\log(1 - D(x, G(x,z))],$$

Fig 3.1 Conditional Loss Function | Image from [13]

Considering the grayscale picture as x, the input noise as z, and the two-channel output we need from the generator as y, (it could also represent the two shade channels of a actual photograph). Also, G is the generator version and D is the discriminator. We can test the importance of the discriminator conditioning, we will compare this to an unconditional version in which x does not have to be observed by the discriminator:

$$\mathcal{L}_{GAN}(G,D) = \mathbb{E}_{y}[\log D(y)] + \mathbb{E}_{x,z}[\log(1 - D(G(x,z))].$$

Fig 3.2. Image from [13]

3.2.1 Optimizing the Loss Function:
The earlier loss function helps to produce good-looking colorful images that seem real, but to further help the models and introduce some supervision in our task, we combine this loss function with L1 Loss (you might know L1 loss as mean absolute error) of the predicted colors compared with the actual colors:

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x,z)\|_1].$$

Fig 3.3 L1 Loss | Image from [13]

If we use L1 loss alone, the model still learns to colorize the images but it will be conservative and most of the time uses colors like “gray” or “brown” because when it doubts which color is the best, it takes the average and uses these colors to reduce the L1 loss as much as possible (it is similar to the blurring effect of L1 or L2 loss in super resolution task). Also, the L1 Loss is preferred over L2 loss (or mean squared error) because it reduces that effect of producing gray-ish images. So, our combined loss function will be:

$$G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G,D) + \lambda \mathcal{L}_{L1}(G).$$

Fig 3.4 Combined Loss function to be optimized [13]

where λ is a coefficient to balance the contribution of the two losses to the final loss (of course the discriminator loss does not involve the L1 loss).

3.3 Neural Network Architecture

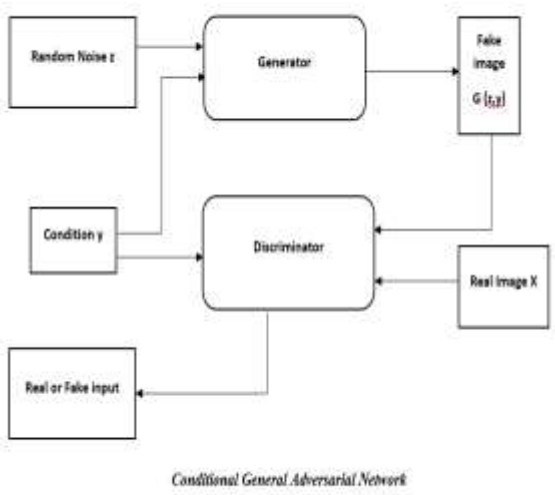

Fig 3.5 CGAN

There are two competing neural network models in Generative Adversarial Nets (GAN). The generator takes the data and creates a fictitious image. The discriminator takes images from both the generator and the label, as well as grayscale or edge-only input, and tries to figure out which pair of photos contains the genuine coloured image. This is depicted in Figure 4.5. During training, the discriminator and the generator are engaged in a continuous game. The generator can produce more realistic photos with each

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 6, Issue 11, ISSN No. 2455-2143, Pages 203-212
Published Online March 2022 in IJEAST (http://www.ijeast.com)

iteration, while the discriminator improves its ability to discern between actual and false photos. The goal is to train a generator to be indistinguishable from real data by training both models together in a minimax method.
In our setting, we are using the L*a*b color space model instead of the RGB to train the model.

3.4RGB vs. L*a*b
As you may know, when we import an image, we obtain a rank-3 (height, width, colour) array with the colour data for our image on the last axis. These statistics describe colour in RGB colour space, with three values showing the amount of Red, Green, and Blue in each pixel. You can see in the next image that we have blue colour on the left half of the "main image" (the leftmost image), thus that part of the blue channel of the image has greater values and has turned dark.

Fig 3.6 Red, Green, Blue channels of an image | Image from Stack

Each pixel in the L*a*b colour space has three numbers, however these numbers possess distinct meanings. The first value (channel), L, encodes the Lightness of each pixel, and it emerges as a black and white image when visualised (the second image in the row below). The *a and *b channels measure the number of green-red and yellow-blue in each pixel. The goal of Lab space is to isolate the brightness (L channel) from the chroma information (channels a and b), while also accounting for the non-linear corrections made by the human brain to the linear signal received from the retina. CIE XYZ space, which represents the physiological response of three of the four types of photo-sensitive cells in the retina, is used to create lab space (the cones). The L*a*b channels are visualized in the image shown below:

Fig 3.7 Lightness, *a, and *b channels of Lab color space for an image | Image from Rodrigo Berriel

When training a colorization model, we should feed it a grayscale image and expect that it will colourize it. When we use L*a*b, we can feed the model the L channel (which is the grayscale image) and ask it to forecast the other two channels (*a, *b), and then appending all of the channels to create our colourful image. However, if you have to use RGB, you'll have to convert the image to grayscale first, then feed the grayscale image to the model and hope it can forecast three numbers for you, which seems to be a more challenging and unreliable process due to several higher probable combinations of three numbers opposed to two.
Assuming every number has 256 variations (the real number of options in an 8-bit unsigned integer image), estimating three numbers for each pixel involves selecting between 2563 possibilities, or more than 16 million possibilities, however estimating two numbers involves deciding from around 65000 choices.

3.5 Generator Model
In our Generator model we define the U-Net model, which was originally created to segment images. A convolutional neural network with an encoder–decoder structure is known as U-Net. The input images are down sampled gradually through a sequence of convolutions eventually till they hit a constriction layer, which comprises a condensed learned depiction of the images. The images are continuously up-sampled after the constriction until they reach the desired output proportions. Also included are skip connections that connect the down-sampling path's outputs to the up-sampling path's outputs. They aid the flow of low-level data through the network, which is hindered by the bottleneck layer. Seven convolutional blocks make up both the encoder and decoder.

3.5.1 U-net Architecture
Limited information may be routed over the network using a U-Net model. Simply disconnecting the skip connectors in the UNet creates the encoder-decoder. In our tests, the encoder-decoder was unable to learn to create realistic pictures. The benefits of the U-Net need not seem to be limited to conditional GANs, if both U-Net as well as the encoder-decoder are trained with an L1 loss, the U-Net again outperforms the encoder-decoder.

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 6, Issue 11, ISSN No. 2455-2143, Pages 203-212
Published Online March 2022 in IJEAST (http://www.ijeast.com)

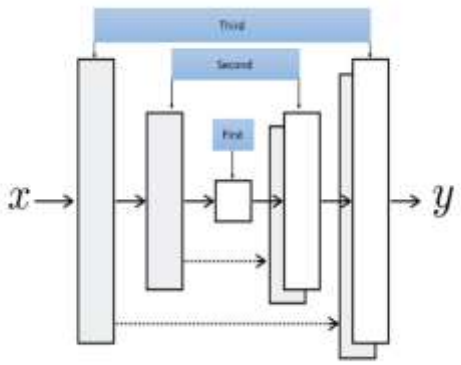

Fig 3.8 The U-net Architecture | Image from [13]

The blue rectangles represent the sequence in which the system's associated modules are built. The constructed U-net will have more levels than this image portrays, but it is enough to contribute to an understanding. Additionally observe that we will be going 8 layers down, so if we start with a 256 by 256 image, we will obtain a 1 by 1 ($256 / 2^8$) image in the middle of the U-Net, which will then be up-sampled to generate a 256 by 256 image (with two channels).

3.6 Discriminator Model

Our discriminator's construction is rather simple. A model is created by stacking Conv-Batch Norm-Leacky ReLU blocks to determine whether the input image is real or artificial. It's worth noting that neither the first nor the last blocks employ normalisation, and the last block lacks an activation function.

3.6.1 Patch-Discriminator

PatchGAN is a convolutional neural network that serves as a discriminator. Generally, discriminators assign a single possibility to the entire image, indicating whether it is real or artificial. Patch GAN, on the other hand, divides the image into NxN patches and generates a probability matrix for every patch. It thus enables the discriminator to provide more useful feedback. One aspect of the image may be realistic, while the other one may need to be improved. Four convolutional blocks constitute the discriminator. Except for the penultimate one, which is sigmoid, all activations are LeakyReLU. PatchGAN's receptive field is 70 by 70 pixels, following industry common procedures. The output form of the model is 30 by 30, however this does not indicate that our patches are 30 by 30. When you compute the receptive field of each of these 900 (30 x 30) output values, which in our instance will be 70 by 70, you get the actual patch size.

IV. TRAINING STRATEGY

4.1 Dataset

The COCO image data was utilised to train the image colorization model (Common Objects in Context) 2017.Despite the fact that the COCO dataset is intended for computer vision tasks like classification and detection and incorporates supervised labels, none of them were employed. There are 83 thousand images in the training subset from 80 separate classes, comprising people, vehicles, food, animals, random objects, and much more. The COCO dataset contains photos of commonplace scenes. As a result, broad world knowledge can be used to train a colorization model.Despite the fact that most current colorization models are trained on ImageNet, which contains 1.2 million training photos, due to technology and time constraints, we train on a significantly smaller dataset. This choice complicates comparability with several other models. However, we can still assess the progress by contrasting the results with those obtained using a model which does not use colorization as a substitute problem.
We extract the first 4000 images to train the model and the last 1000 images for validation. The images are enlarged and flipped horizontally, and then an RGB image is used as an input, then convert it to the Lab colourspace. Furthermore, split the first (grayscale) and colour channels as the inputs and targets for the models, accordingly.

Fig 4.1 Examples images and masks from the COCO dataset 2017.

4.2 Forward and Backward function methods

From input Tensors, the forward function computes output Tensors. The reverse function takes the output Tensors' gradient with respect to a scalar value and computes the input Tensors' gradient with respect to the same scalar value. We use the functions to specify our generator and discriminator, as well as to initialise them. Finally we create our two loss functions, as well as the generator and discriminator's optimizers.Then, using the backward method, we train the discriminator by feeding the discriminator the false images generated by the generator and classifying them as false. The discriminator is then fed a batch of real photos from the training set, which are classified as real. We take the average of the two losses for fake and actual, then perform backward on the total loss.
We can now begin to train the generator. We feed the discriminator the fake image and try to trick it by assigning actual labels to it and computing the adversarial loss in the backward technique. As previously stated, we use L1 loss to compute the distance between the predicted and target two channels, multiply this loss by a coefficient (in our case,

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 6, Issue 11, ISSN No. 2455-2143, Pages 203-212
Published Online March 2022 in IJEAST (http://www.ijeast.com)

100) to complement the two losses, and then add this loss to the adversarial loss. The backward method of the loss is then called.
The Leaky Rectified Linear Unit, or Leaky ReLU, is a kind of activation function derived on the ReLU, but instead of a flat slope for negative values, it has a tiny slope. The slope coefficient is calculated before to training, rather than being learned during training.

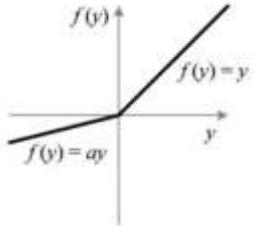

Fig 4.2 Leaky ReLu Activation Function.

The training of a neural network is done via supervised machine learning at a high level. The model is fed a batch of data, and the model's predictions are evaluated in terms or ground truth values for the inputs. We are initializing the weights of our model with a mean of 0.0 and standard deviation of 0.02 which are the proposed hyper parameters.

4.3 Final Model
After training the model for upto 50 epochs, the images are resized to 256x256 and are fed to a batch of 250.

Loss_d_fake	0.55053
Loss_d_real	0.55889
Loss_d	0.55471
Loss_g_GAN	1.22825
Loss_g_L1	6.42693
Loss_g	7.65519

Table 4.1 Evaluating scores of the PatchGAN colorization model.

Here we can observe that the discriminator has successfully classified real image at an accuracy rate of 55 % .The loss function L1 is still rather high, indicating that our model has to be improved further. The loss does not punish mismatch between the input and output in this situation; all that concerns is that the result appears realistic. The performance of this version is poor.

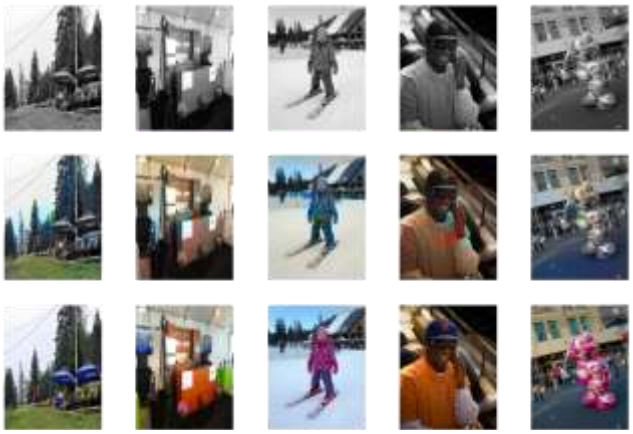
Fig 4.3 Examples of colorization in the first training strategy.

Although our baseline model includes a rudimentary grasp of some of the most frequent things in photos, such as the scenic landscapes, the output is unappealing, and it can't seem to decide what colour specific items should be. It also has some colour spillover effects and a circular pattern blob of colour in the middle of the first image in the second row, which isn't nice. So, it appears that we won't be able to obtain decent results with this method with this limited dataset. As a result, we alter our strategy.

4.4 A new proposed experimental setup
To circumvent the dilemma of "the blind leading the blind" in the GAN game, where neither the generator nor the discriminator knows anything about the job at the start of training, I chose to pretrain the generator individually in a supervised and deterministic manner.

There are two pretraining phases :
1. The pretrained model for classification constitutes the baseline of the generator which is the down sampling channel.
2. Colorization with L1 loss will be pre-trained over the whole generator.

In practice, we are going to incorporate a pretrained ResNet18 as the foundation of my U-Net, and we'll train the U-Net on our training set with just L1 Loss to complete the final step of pretraining. Then, like in the preceding part, we'll go on to the combination adversarial and L1 loss.

4.5Constructing a new generator
In the new generator setting, we make use of the Dynamic U-Net module which is used for our Semantic Segmentation. We load the ResNet18 architecture's pretrained weights and trimming the model to eliminate the last two layers which are the GlobalAveragePooling and a Linear layer. The backbone is then used by DynamicUnet to create a U-Net with the required output channels which is 2 in our setting and an input size of 256.

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 6, Issue 11, ISSN No. 2455-2143, Pages 203-212
Published Online March 2022 in IJEAST (http://www.ijeast.com)

The model is pre-trained for 20 epochs. Using GPU-powered Collab Notebooks, the entire model training procedure took up to 48 hours, with the pretraining step taking 6 hours. We train the entire model using the newly proposed Generator model, using the same steps as previously.I'm importing the generator's stored weights first, then utilising this model as the generator in our previous module which avoids the generator from being arbitrarily initialised.With a learning rate of 0.002, Adam was applied as an optimizer.

V. RESULTS

The multilabel classification model was trained in two different ways. The first one was trained from scratch using a Gaussian distribution with mean 0 and standard deviation 0.02, thus it will serve as our baseline. Except for the new classifier's weights, the second model was started with the learnt weights from the picture colorization model. Binary cross-entropy loss, an Adam optimizer, and a batch size of are used in both models. With a learning rate of 2 $x10^{-4}$, the initial model was trained for 50 epochs. The second model underwent two periods of training. The colorization weights were fixed for the first three epochs, and a learning rate of 1 x 10^{-4} was utilised.

Model initialization	**Image classification (Accuracy)**	**Image colorization (Accuracy)**
Baseline	55%	65%
Pretrained Colorization model	63%	78%

Table 5.1 Evaluation of the accuracy rate of the final model.

5.1Using the pretrained U-Net with and without adversarial training to compare the results

The source of noise in the prior model architecture was the dropout layers. However, after deploying the U-Net, I investigated and have found that there are not dropout layers. Is it possible that the generator can have a creative influence on the output if there is no noise and whetherit is conceivable that the grayscale picture that the generator receives also acts as noise were some of the questions we may ask.

5.2Analysis

The conditional GAN can still function without drop outs, however the outputs will be more predictable due to the absence of noise; nevertheless, there will still be significant information in the initial monochrome image for the generator to create appealing results.Several of the interesting things I uncovered throughout my investigations is that the U-Net we constructed with the ResNet18 backbone is already efficient at colourizing pictures having simply pretraining with L1 Loss. However, the model is particularly cautious, advising users to choose greyish hues when uncertain as to what the entity is and what colour it should be. However, for frequent situations in photos such as sceneries, plants, people and so on, it performs exceptionally well.

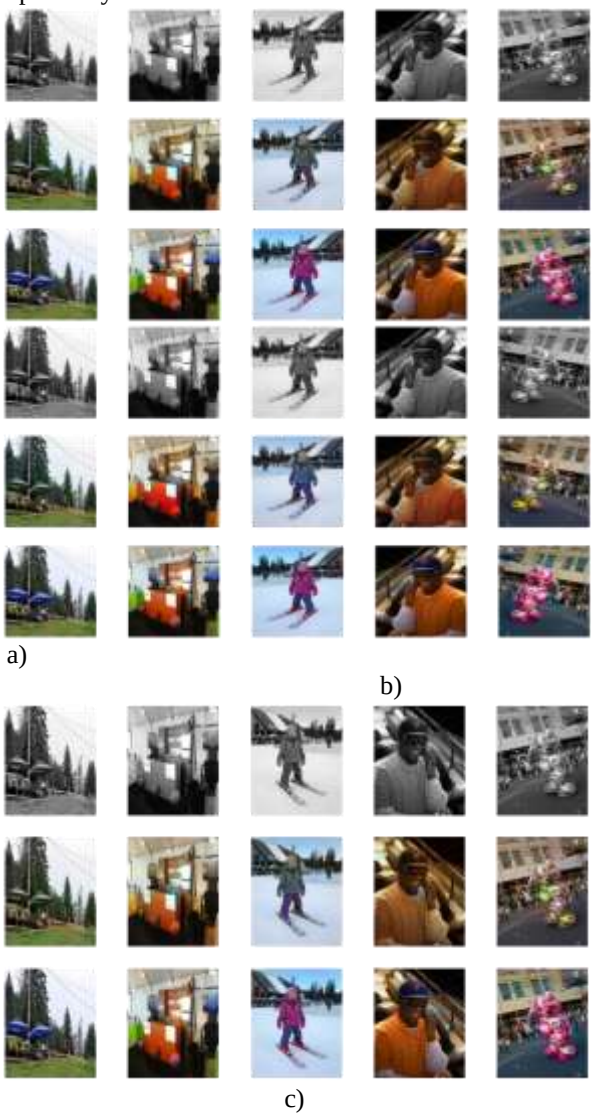

Fig 5.1 Pretrained U-Net without adversarial training

To better illustrate the huge difference that adversarial training produces in our scenario, we present the results of the U-Net without adversarial training and the U-Net with adversarial training.

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 6, Issue 11, ISSN No. 2455-2143, Pages 203-212
Published Online March 2022 in IJEAST (http://www.ijeast.com)

a)

b)

c)

Fig 5.2 Pretrained U-Net with adversarial training.

Loss_d_fake	0.63442
Loss_d_real	0.64528
Loss_d	0.63985
Loss_g_GAN	0.89723
Loss_g_L1	5.88712
Loss_g	6.78434

Table 5.2 Evaluating scores of the Pretrained U-Net with adversarial training.

As you can see, the pretrained U-Net is indeed excellent, but it did not pick colours accurately in many scenarios, so it covers regions with reddish/brown. We can observe that the adversarial training architecture performs more efficient work.

VI. CONCLUSION

Image colorization is a specialised computer graphics activity, but it is also an example of a tough computer vision pixel prediction problem. When we look at several research publications, we notice that the diverse methodologies and strategies used can give results that are indistinguishable from true colour images. Various methodologies, as well as a well-chosen loss function, significantly improve the results. I'd want to express my gratitude to the authors of the excellent works for their contributions.

6.1Future Works

This application may be considered in future works with more success. We've revealed certain colorization restrictions in this paper, which leaves the issue open for further investigation. While conducting this research, the following are some noteworthy future work recommendations:

- In the future, a mix of deep learning algorithmic techniques might improve the system interface for picture and video colorization, demonstrating their efficacy.
- The system works well with organised data like shoes and apparel, but not so well with unstructured data like landscapes.
- Future study for this platform might focus on reducing time-consuming user input while keeping creative control over outcomes, as well as boosting algorithm speed to allow artists to engage with the colorization system in real time.

VII. REFERENCES

[1] A. D. Pyngrope, P. Student, and P. Kumar, “COLORIZATION OF BLACK AND WHITE IMAGES: A SURVEY,” vol. 10, no. 3, p. 3, 2022.

[2] X. Liu et al., “Intrinsic Colorization,” ACM Trans. Graph., vol. 27, p. 152, Dec. 2008, doi: 10.1145/1457515.1409105.

[3] R. Ironi, D. Cohen-Or, and D. Lischinski, “Colorization by Example,” Jan. 2005, pp. 201–210. doi: 10.2312/EGWR/EGSR05/201-210.

[4] G. Charpiat, M. Hofmann, and B. Schölkopf, “Automatic Image Colorization Via Multimodal Predictions,” Sep. 2008, vol. 2008. doi: 10.1007/978-3-540-88690-7_10.

[5] A. Chia et al., “Semantic Colorization with Internet Images,” ACM Trans. Graph., vol. 30, p. 156, Dec. 2011, doi: 10.1145/2070781.2024190.

[6] R. Gupta, A. Chia, D. Rajan, E. Ng, and Z. Huang, “Image colorization using similar images,” Oct. 2012, pp. 369–378. doi: 10.1145/2393347.2393402.

[7] Z. Cheng, Q. Yang, and B. Sheng, “Deep Colorization,” Apr. 2016.

[8] S. Treneska, E. Zdravevski, I. Pires, P. Lameski, and S. Gievska, "GAN-Based Image Colorization for Self-Supervised Visual Feature Learning," Sensors, vol. 22, Feb. 2022, doi: 10.3390/s22041599.

[9] S. Koo, "Automatic Colorization with Deep Convolutional Generative Adversarial Networks," 2016. https://www.semanticscholar.org/paper/Automatic-Colorization-with-Deep-Convolutional-Koo/ca769bc02cb1b74a160d606fbb171afb13d0d615 (accessed Mar. 23, 2022).

[10] P. Vitoria, L. Raad Cisa, and C. Ballester, ChromaGAN: An Adversarial Approach for Picture Colorization. 2019. doi: 10.13140/RG.2.2.33484.97927.

[11] S. Anwar, M. Tahir, C. Li, A. Mian, F. Khan, and A. Muzaffar, "Image Colorization: A Survey and Dataset," Aug. 2020.

[12] R. Zhang et al., "Real-time user-guided image colorization with learned deep priors," ACM Trans. Graph., vol. 36, no. 4, pp. 1–11, Jul. 2017, doi: 10.1145/3072959.3073703.

[13] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-Image Translation with Conditional Adversarial Networks," arXiv:1611.07004 [cs], Nov. 2018, Accessed: Feb. 03, 2022. [Online]. Available: http://arxiv.org/abs/1611.07004

[14] M. Mirza and S. Osindero, "Conditional Generative Adversarial Nets," Nov. 2014.

[15] M. R. Joshi, L. Nkenyereye, G. P. Joshi, S. M. R. Islam, M. Abdullah-Al-Wadud, and S. Shrestha, "Auto-Colorization of Historical Images Using Deep Convolutional Neural Networks," Mathematics, vol. 8, no. 12, p. 2258, Dec. 2020, doi: 10.3390/math8122258.

[16] Y. Morimoto, Y. Taguchi, and T. Naemura, "Automatic colorization of grayscale images using multiple images on the web," Jan. 2009. doi: 10.1145/1599301.1599333.

[17] A. Deshpande, J. Rock, and D. Forsyth, "Learning Large-Scale Automatic Image Colorization," Dec. 2015, pp. 567–575. doi: 10.1109/ICCV.2015.72.