**·Model/Architecture:**

GANs are generative models that consists in two different networks: the Generator and the Discriminator. The generator is trained to produce outputs that cannot be distinguished from “real” images by an adversarially trained discriminator, which is trained to do as well as possible at detecting the generator’s “fakes”.

We implemented a pix2pix GAN type model for our colorization model. In this image-to-image translation model, we introduce our input images only using the Lightness L channel following the CIELAB color space and we receive as output the a and b channels from the image.

Afterwards we condition our discriminator with the concatenation of both the input and output image. This, in essence, is the same as passing through the output LAB image with the real LAB image.

In the original pix2pix paper the authors used dropout to create some stochasticity and have a variety of outputs for a given input. We didn't implement the GAN as such, and therefore our model is deterministic in nature.

We tried a variety of different parameters during our training. We used two different generators: an Unet and a Resnet. In addition, we have used three different discriminators: A “Critic”, a “Patch Discriminator” and a “Pixel Discriminator”.

The Critic discriminator consists of several convolutional layers that downsample the input images, and on the latest layer we add a fully connected layer. Is important to notice that this discriminator doesn’t have a softmax as we prefer working with the output logits.

The Patch discriminator, or PatchGan, consists on a serie of convolutional layers. It only penalizes structure at the scale of patches. This discriminator tries to classify if each N ×M patch in an image is real or fake. This is advantageous because a smaller PatchGAN has fewer parameters, runs faster, and can be applied to arbitrarily large images. The area of the patch can be very small, and still produce high quality results.

The Pixel Discriminator is a PatchGan of size 1x1, with each convolution having a stride of 1. The prediction is only on pixel level and it works surprisingly well on our case.

**·Losses:**

The main advantage of using GAN over other more traditional approaches is that it learns a loss that adapts to the data, they can be applied to a multitude of tasks that traditionally would require very different kinds of loss functions. We have proven that a distance-based loss function (such as L1, L2, Hubber, etc…) is not well suited for a colorization task, as the output is averaging all the plausible and realistic cases.

In our model, we tried three different losses: “Vanilla”, “LSGAN” and “WGAN-GP”.

**·Results:**

Using Vanilla Loss, an Unet generator and a Pixel Discriminator we have trained our GAN for 30 epochs. We have used a small dataset of Celebrities faces on our colorization problem.

The results on the image below show the losses from the discriminator and from the generator. We have split further the discriminator loss between the fake Discriminator loss and the real Discriminator loss.

A graph of different colored lines

Description automatically generated

We can see from the image above that as soon as the Discriminator becomes better, the Generator becomes better also. This is typically the case with a well-designed minimax optimization problem. Additionally, the lack of a proper evaluation metric makes GANs harder to train and know when we should stop. The images below show the different values of the L2 loss, the Peak-To-Noise-Ratio and the Structural Similarity Index our model is likely to be overfit.

A graph of a graph showing the loss of a person

Description automatically generatedA graph with blue and orange lines

Description automatically generatedA graph with blue and orange lines

Description automatically generated

We believe the cause is that even if our GAN losses are oscillating, the GAN loss has plateaued, while the other losses might become more important. We believe that we should fine-tune the lambda parameter in our model to handle the relative importance of each loss. We might be able to vary the learning rate of our model to reach a different result.

Additionally, increasing the dataset size will likely not solve the weighting loss problem, but it will likely solve the overfitting part.

A collage of a person

Description automatically generated